A NOVEL DECOMPOSITION STRUCTURE FOR ADAPTIVE SYSTEMS

By

Wan, Kwok Fai

(尹國輝)

A thesis presented to the Chinese University of Hong Kong
in partial fulfillment of the degree of Doctor of Philosophy

Department of Electronic Engineering
The Chinese University of Hong Kong
Shatin, N.T., Hong Kong

May, 1995

ACKNOWLEDGEMENTS

The author wishes to express his heartfelt gratitude to his supervisor, Dr. P.C. Ching, for his many encouragements, excellent guidance and inspiring advice throughout the research work and during the preparation of this thesis.

The author is also grateful to Dr. K. C. Ho for his valuable comments and discussions.

ABSTRACT

Considerable accomplishments have been achieved for research in the area of adaptive signal processing, specifically in the design of adaptive filter structure and development of efficient weight-adjustment algorithms. Applications of adaptive signal processing can be found in different fields such as communication, radar, sonar, seismology, mechanical design, automatic control, biomedical electronics and many others.

The least mean square (LMS) algorithm, based on the steepest-decent method, is by far the simplest and most widely used adaptive algorithm. However, one of the major deficiencies of the LMS algorithm is the relatively slow convergence speed which restricts its applications in many practical situations. Many research efforts have been spent on improving the performance of the LMS algorithm, and the split algorithm is found to be efficient in providing better adaptation characteristics. The idea of split-path adaptive filtering is to reduce the ill-condition of the system by dividing the original transversal filter into two linear phase subfilters connected in parallel, one with antisymmetric and another with symmetric property. When appropriate convergence factors are chosen for the two filters, the system performance, specifically the convergence speed, can be enhanced significantly.

Despite the fact that the split structure adaptive filtering has been demonstrated to be a useful technique, many inherent properties still need rigorous analysis and its generalization property and the relationship with transform domain adaptation require further investigation. Precluding the restriction that the technique can only be applied for non-symmetric system is also an interesting topic.

Furthermore, it is also interesting to study the possibility of adapting a split-path filter structure in transform domain in order to obtain further improvement of an adaptive system.

In this thesis, a detailed theoretical analysis on the properties of the split-path adaptive algorithm is given and subsequently a split-path median LMS algorithm is proposed which has distinct merits in suppressing impulsive noises and tracking the edges of incoming signals.

A novel technique to apply the split algorithm to a symmetric/antisymmetric adaptive FIR filter has also been devised. It is then shown that an $M=2^L$-coefficient transversal filter can be decomposed, by L steps splitting operation, into M single parameter adaptive subunits connected in parallel, which leads to a modular form for the multi-stage split-path adaptive algorithm. The analysis is general and is suitable for most gradient-based adaptive algorithms with particular emphasis on easy implementation of the LMS method.

The split-path structure has been applied to transform domain adaptive filtering and an approach to further improve the performance behaviour in transform domain adaptation by introducing variable convergence factor is suggested. In addition, a practical optimal convergence factor tracking method for the transform domain split-path adaptive system is derived that enables a fast convergence speed up.

Finally, the inherent characteristics of the split-path operation, especially its capability of data compression is investigated thoroughly. This analysis results in the establishment of a unification between split-path filtering and discrete Walsh transform adaptation. A new Walsh function through splitting matrices is proposed which has the advantage of implementing efficiently the fast discrete Walsh transform algorithm.

CONTENTS

FIGURE CAPTIONS

TABLE CAPTIONS

Glossary of Symbols and Abbreviations

Symbols

∇	gradient operation
$\|A\|$	norm of vector A
λ	eigenvalue of input autocorrelation matrix
γ	eigenvalue spread
μ	time domain convergence factor
ε	time domain mean square error
η	transform domain convergence factor
Φ	transform domain correlation matrix
ε^{sp}	split-path transform domain mean square error
ε^{t}	transform domain mean square error
*	denotes optimal value
cov{...}	covariance
det(**A**)	determinant of matrix A
diag{…}	diagonal matrix with elements … on main diagonal
E	expected value operation
$\boldsymbol{I}$	identity matrix
$\boldsymbol{J}$	anti-diagonal identity matrix
$\boldsymbol{R}$	time domain correlation matrix
^{T}	transpose
$\boldsymbol{U}$	splitting matrix
$\boldsymbol{V}$	orthogonal transform matrix

Abbreviations

ALMS	averaged least mean square
AR	autoregressive
DCT	discrete cosine transform
DFT	discrete Fourier transform
DHT	discrete Hartly transform
DST	discrete sine transform
DWT	discrete Walsh transform
FFT	fast Fourier transform
FIR	finite impulse response
FKLT	fast Karhunen-Loeve transform
i.i.d	independent identically distributed
IIR	infinite impulse response
KLT	Karhunen-Loeve transform
LMS	least mean square
MLMS	median least mean square
MSE	mean square error
OS	order statistic
RLS	recursive least square
SCT	symmetric cosine transform
SM	splitting matrix
SPMLMS	split-path median least mean square
TDAF	time domain adaptive filter
TDL	tapped delay line
TRAR	transform domain adaptive filter
TRSPAF	transform domain split-path adaptive filter

Chapter 1 ADAPTIVE SIGNAL PROCESSING AND ITS APPLICATIONS

1.1 Introduction

The usual optimal Wiener method [Wiener, 1949], [Kailath, 1974] of estimating a signal corrupted by additive noise is to pass it through a filter that tends to suppress the noise while leaving the desired signal relatively unchanged. The Wiener filter here acts as a correlation cancellor. If certain statistical noises and desired signals are, however, in any way uncorrelated, the filter will cancel from the output any such uncorrelations.

The filtering methods used for the above application can be fixed processing or adaptive processing techniques. The design of fixed filters is based on prior knowledge of both the signal and noise, or at least their statistical estimates, in advance. If the environment changes, the optimum filters must be redesigned to match to the data being processed by it.

In many instances, however, the complete range of input conditions may not be known exactly, or even statistically; or the conditions may change from time to time. We do not know how often to redesign an optimum filter. The fixed filters are for the most part inapplicable because the correlation and cross correlation functions of the desired signal and reference inputs are generally unknown and often variable with time.

In such circumstances, the adaptive implementations would provide superior performance compared with a system of fixed design.

Adaptive systems are learning systems in the sense that they "learn" their operation environment. The essential and principal property of the adaptive system is its time-varying, self-adjusting performance. The adaptive system can automatically adjust its filter parameters even its structure and continually track the environment by adaptive algorithms to optimize some cost functions or performance criterions. In general, the convergence speed and mean square error (MSE) of an adaptive system are the major considerations.

In an adaptive implementation, a filter must has a finite number of parameters or filter weights. These parameters are adjusted until they converge to their optimal values. A typical adaptive filter is depicted in Fig. 1.1. At each time instant k, the current values of the parameters are used to perform the filtering operation. The adaptive algorithm continuously monitors the error signal $e(k)$ between the actual output $y(k)$ and the desired signal $d(k)$ and attempts to minimize the power of the output error $\mathrm{E}[e^2(k)]$, or equivalently tries to decorrelate $e(k)$ from the input $x(k)$. Here, computed error $e(k)$ is used by the adaptation part of the algorithm to change the parameters in the direction of their optimum values. As processing of the signal $d(k)$ and $x(k)$ takes place and the filter gradually learns the statistics of these signal, its parameters gradually converge to their optimum values given by Wiener solution.

Clearly, the input statistics must remain unchanged for at least as long as it takes the filter to learn it and converge to its optimum configuration. If, after convergence, the input statistics should change, the filter will respond by readjusting its

parameters to their new optimum values, and so on. In other words, the adaptive filter will have the capability of tracking nonstationary changes of the input statistics as long as such changes occur slowly enough for the filter to converge between changes.

1.2 Applications of Adaptive System

Applications for adaptive systems are numerous and can be found in areas such as communications, radar, sonar, seismology, mechanical design, navigation systems, and biomedical electronics. Some typical applications of adaptive system will be described in this section.

1.2.1 Adaptive Noise Cancellation

One of the simplest and most effective adaptive signal processing techniques is adaptive noise cancellation [Widrow 1975]. There are many applications of adaptive noise canceling, such as adaptive sidelobe cancellation, acoustic noise cancellation [Harrison 1986] [Müller 1986] and ghost cancellation in television [Ciciora 1979]. A configuration of adaptive noise cancellor is shown in Fig. 1.2 where the desired signal $d(k)$ is corrupted by an additive noise $n_1(k)$. Assume a second noise $n_2(k)$, which is in any way correlated with the noise component in $x(k)$, is also available, the adaptive filter will continuously adjust its parameters to produce an output $y(k)$ until such correlations are canceled from the system output $e(k)$. In the procedure, $y(k)$ resembles $n_1(k)$, and finally output $e(k)$ will only consist the desired signal $d(k)$.

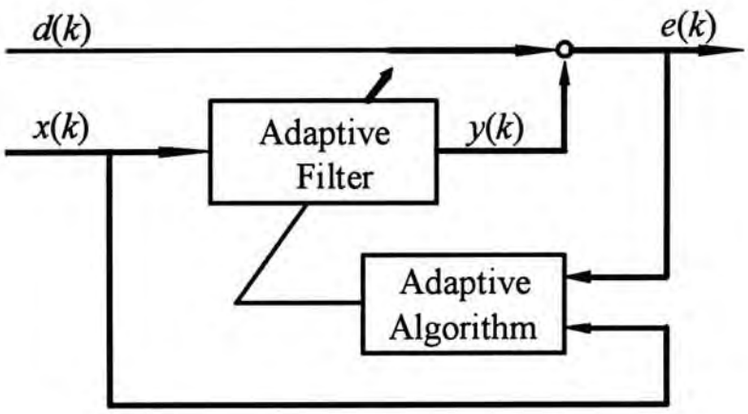


Fig. 1.1 An adaptive Filter.

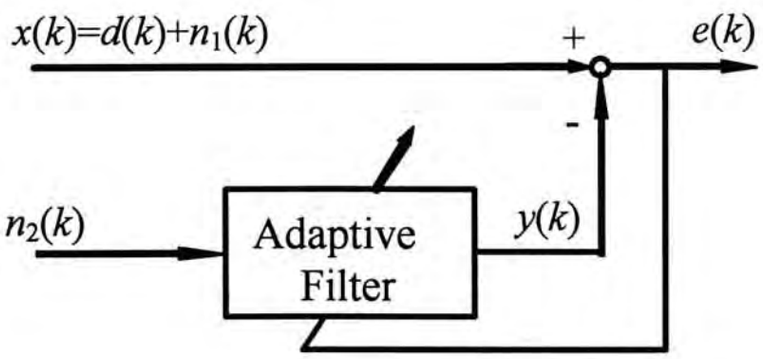


Fig. 1.2. Adaptive noise cancellation.

1.2.2 Adaptive Echo Cancellation

Adaptive echo cancellation is a special interference cancellor that is widely used to suppress the echo in the telephone network [Messerschmitt 1984]. Fig. 1.3 is a block diagram of a simple telephone network. When speaker A's speech reaches B, it will also echo back a "talker echo" to A, as though it were B's speech. To avoid such unwanted interference, an adaptive filter is inserted in the circuit nearby terminal B. The adaptive filter here serves as an echo chancellor, its output, $y(k)$, is used to cancel A's echo. When speech signal $x(k)$ is transferred to speaker B, it is at the same time fed to adaptive filter as well. That is, adaptive filter should adjust itself to produce an estimate of A's echo in real-time and subtract this predicted echo from the transmission path signal, which contains the actual echo, $d(k)$.

1.2.3 Adaptive Line Enhancement

An adaptive line enhancement is actually a self-tuning filter [Widrow 1975] [Treichler 1979] [Zeidler 1978] used in the case when there is only one signal $x(k)$ available which is contaminated by noise. Fig. 1.4 shows a typical adaptive line enhancer. Suppose the signal $x(k)$ consists of two components: a narrowband component $h(k)$ that has long-range correlations such as sinusoid or some other periodic signals, and a broadband component $n(k)$ which will tend to have short-range correlations. The signal $x(k)$ itself provides as the reference signal and its delay, $x(k-\Delta)$, is taken as the input of the adaptive filter. If the delay Δ is selected longer than the effective correlation length of broadband component $h(k)$ and shorter than the

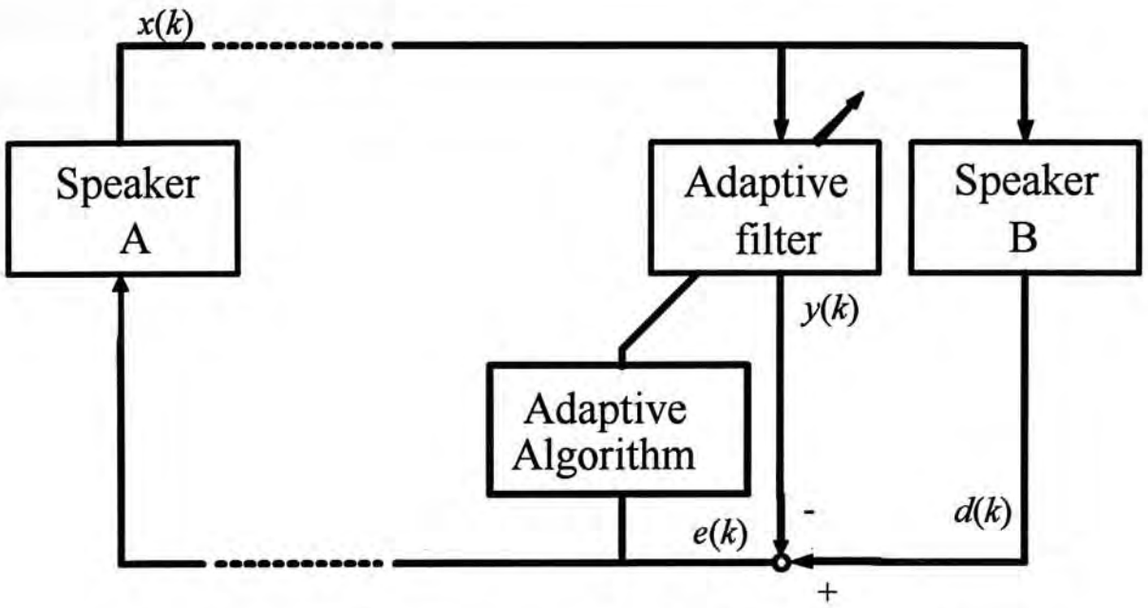


Fig. 1.3. Adaptive echo cancellation in telephone network.

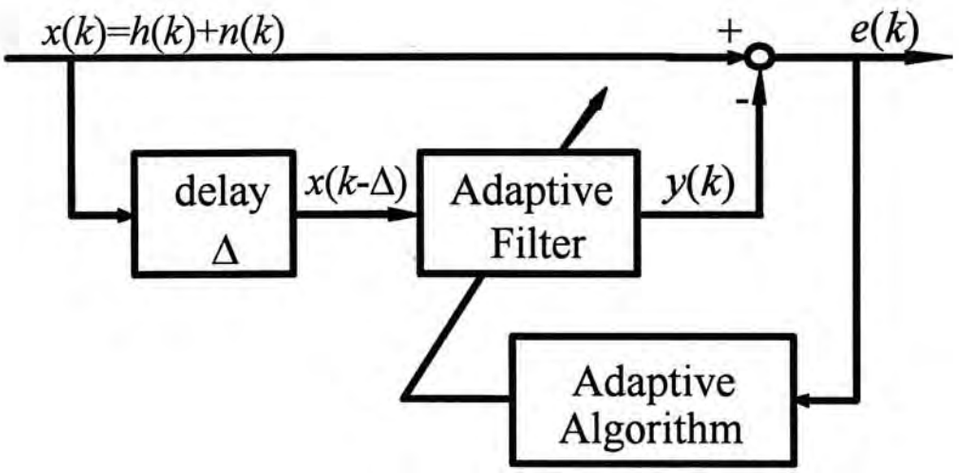


Fig. 1.4. Adaptive line enhancement.

correlation length of the narrowband component $n(k)$, the adaptive filter will not be able to respond to narrowband signal $h(k)$ but respond to the broadband noise component $n(k)$. In such way the estimate of the noise $n(k)$ will appear at the output of adaptive filter which is used to cancel the noise in the original input $x(k)$.

1.2.4 Adaptive Linear Prediction

A linear predictor is a special case of adaptive line enhancement with the delay Δ=1 (Fig. 1.5). Many adaptive signal processing techniques are based on the solution of the linear prediction problem. In the system, the input $x(k)$ is taken as the desired signal and a delayed version of the input is sent to the adaptive processor. In adaptation, the adaptive filter will try to "predict" current input signal using the information given by past data. That is to generate an output in order to have $y(k)$ cancel $x(k)$ and drive $e(k)$ toward zero. The adaptive predictor may be considered as a kind of adaptive whitening filter. In fact, when adaptation takes place, the parameter of the adaptive filter is really the estimate of an autoregressive model. Linear prediction is used in signal encoding, noise reduction, time series modeling, speech processing, spectral estimation [Haskew 1973] [Makhoul 1973], and many other adaptive signal processing applications.

1.2.5 Adaptive System Identification

System identification is an important topic in automatic control system [Åstron 1984]. An adaptive system identification scheme can be simplified as in Fig. 1.6. The purpose of the system is to determine the changing structure and/or parameter of the unknown plant. Here a broadband signal is sent through the unknown plant as well the adaptive filter and produce outputs $d(k)$ and $y(k)$, respectively. By minimizing the difference of two outputs, the parameters in filter are adjusted by adaptive algorithm and will approach to, that is to identify, the transfer function of the unknown plant. Adaptive system identification can be used in system modeling [Friedlander 1982] and adaptive control to model an unstationary plant which is varying slowly [Goodwin 1977] [Åstron 1984].

Other than the above applications, the adaptive systems have also been widely used to resolve problem of inverse modeling, channel equalization and many others. Adaptive equalization could be used to deconvolve the effects of a transducer, a communication channel or to produce an inverse model of an unknown plant [Lucky 1966] [Gersho 1969] [Satorius 1981] [Ungerboeck 1972]. It is also applicable in the design of digital filters and adaptive control system [Widrow 1985]. Adaptive beamforming or adaptive array processing is yet another area that has attracted a lot of research attention. [Haykin 1980]. Adaptive beamformers are truly learning and self-optimizing systems which can adjust their beam patterns to improve the signal-to-interference ratio [Widrow 1967] [Frost III 1972] [Applebaum 1976] and have found practical applications in radar and sonar.

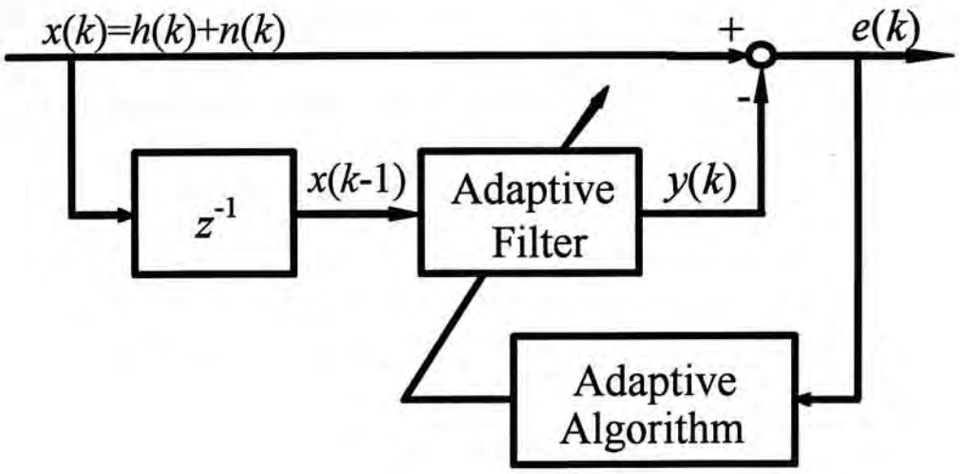


Fig. 1.5. Adaptive linear predictor.

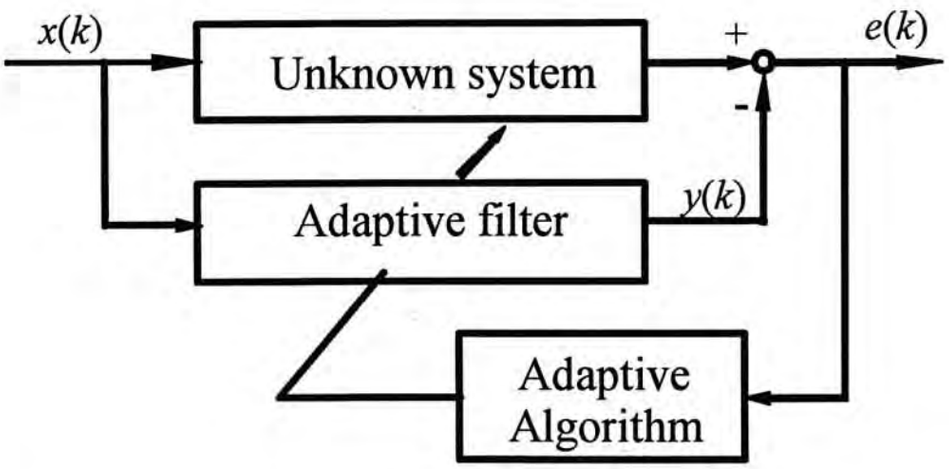


Fig. 1.6. Adaptive system identification.

1.3 Algorithms for Adaptive Systems

Although in theory, both FIR and IIR structures can be used for adaptive filters, the stability problems and the local optimum points that the IIR filters exhibit make them less attractive for such an application. Hence, until further research makes IIR filters a better choice, only the FIR filters are used in adaptive algorithms of practical applications.

In the realization of adaptive systems, different kinds of adaptation algorithms can be employed, such as the Widrow-Hoff least mean square (LMS) algorithm, the conventional recursive least squares (RLS) algorithm, the fast RLS algorithms, and the adaptive lattice algorithms. A large number of literatures have been published to discuss the characteristics of various adaptive algorithms and their applications. The following basic issues in any adaptive algorithm are:

1. The learning or convergence speed of the algorithm.
2. The computational complexity of the algorithm.
3. The misadjustment errors or numerical accuracy of the algorithm.
4. The stability and robustness of the algorithm.

The convergence speed is an important factor because it determines the maximum rate of change of the input nonstationarities that can be usefully tracked by the filter. The computational complexity refers to the number of operations required to update the filter from one time instant to the next.

The LMS algorithm, based on the steepest-descent method, is the most widely used algorithm because of its simplicity and ease of computation. It provides a gradual, iterative, minimization of the performance index [Widrow 1976]. The

adaptive weights are not optimal at each time instant, but only after convergence. The restriction of the LMS is its slow in convergence and being applicable primarily to nonrecursive linear filters [Caraiscos 1984].

The RLS algorithms are the time-recursive analogs of the block processing method of linear prediction and FIR Wiener filtering. Because of their fast convergence they have been proposed for use in fast start-up channel equalization [Godard 1974] [Chang 1971] [Mcwhirter 1983], real-time system identification applications [Goodwin 1977] [Åstron 1984], and rapidly varying adaptive arrays [Brennan 1974]. Their major disadvantage is that they require a fair amount of computation ($O(M^2)$, for M-tap filters) per time update.

The recent fast reformulations of RLS algorithms combine the best of the LMS and RLS, namely, the computational efficiency of the former and the fast convergence of the latter. The fast RLS algorithms reduced the computational complexity on the cost of unstable for its performance.

Gradient adaptive lattice is based on a gradient-descent, LMS-like approach applied to the weights of the lattice representations rather than to the weights of the direct form representation. Taking advantage of the decoupling of the successive stages of the lattice, the convergence rate of weights can be fast and is independent of eigenvalue spread of the input covariance matrix.

The table shown below briefly compares the basic performances of several adaptive algorithms [Orfanidis 1988].

algorithm	speed	complexity	stability
LMS	slow	simplest	stable
RLS	fast	complex	stable
Fast RLS	fast	simple	unstable
Lattice	fast	simple	stable

Table. 1.1. Comparison of some adaptive algorithms.

Surely, the LMS is the simplest whilst its main drawback is relatively slow in convergence speed. In practical, the choice of a particular algorithm is very much depended on the requirement of system and operation environments. If the adaptive system is an adaptive linear combiner or a tapped delay line filter, and if the input vector $x(k)$ and the desired response $d(k)$ are available at each iteration, the LMS algorithm is generally the best choice for many different applications of adaptive signal processing, particularly when the environment is changing slowly.

1.4 Transform Domain Adaptive Filtering

In addition to the researches in conventional time domain LMS algorithms, transform domain adaptive filtering is also an attractive area for adaptive signal processing.

In general, it can be shown [Widrow 1976] that for stationary input and sufficiently small convergence factor, the speed of convergence of the algorithm is

dependent on the eigenvalue spread or the ratio of maximum to minimum eigenvalues of the input autocorrelation matrix. This ratio, often referred to as the condition number, is a measure of ill-conditioning of the matrix. Slow convergence rate can be expected when this ratio is large. An approach to accelerate the convergence rate is to somehow transform the input signal $x(k)$ into another signal with the corresponding autocorrelation matrix having smaller eigenvalue spread. This can be achieved by performing the adaptive filtering in some orthogonal transform domain.

The principle of the transform domain adaptation is to whitening the power spectrum of the input signal by prefiltering the input sequence, reduce its eigenvalue spread, then update the parameter in the transform domain. If it is needed, the time domain parameter can easily be obtained from transform domain parameters. For example, discrete Fourier transform (DFT) [Bershad 1979], discrete cosine transform (DCT) [Ahmed 1974], symmetric cosine transform (SCT) [Kitajima 1980], fast Karhunen-Loeve transform (FKLT) [Jain 1976], discrete sine transform (DST) [Wang 1982], discrete Hartly transform (DHT) [Bondyopadhyay 1988], and discrete Walsh transform (DWT) [Beauchamp 1984] has been introduced in adaptive algorithm and are known to have significantly improved convergence speed [Marshall 1989].

1.5 The Motivation and Organization of the Thesis

It is known that the convergence factor and the eigenvalue spread of the input correlation matrix play two important roles in the performance of the adaptive system. Reducing the eigenvalue spread or suitable choosing the convergence factor can improve the performance of the adaptation. Many efforts have been made in this

regard. Some researchers attempt to improve the inherent characteristics of the data sequence to be processed, such as block implementation of adaptive digital filters [Clark 1981] or transform domain adaptation [Narayan 1983]. On the other hand, emphasis is also put on designing new adaptive filter structure in order to cope with different kinds of signal and to reduce computational complexity. Split-path filtering is one of the effective structures for adaptive signal processing [Ho 1991b] [Ho 1992a].

The principle of split-path adaptive filtering is to divide the original transversal filter into two linear phase subfilters connected in parallel, one with antisymmetric and another with symmetric property. When appropriate convergence factors are chosen for the two filters, the system performance, specifically the convergence speed, can be enhanced significantly.

Although the split structure adaptive filtering has been proved to be a useful technique for adaptive signal processing, its general properties including its extensive version and relationship with transform adaptation are not known. Furthermore, it will be very useful to develop a generalized split-path structure that can be combined with the most widely used LMS algorithm to achieve better performance characteristics.

In this thesis, we focus our research on the following aspects:

1. To analyse the properties and performance of the split-path structure with an application to adaptive median filtering.
2. To study the possibility to extend the split-path adaptive filter to a more general form, *viz*, to derive a continuous or a full modular split structure adaptive algorithm.

3. To study the properties and performance of the split-path structure when it is applied to transform domain adaptation, and then to introduce a variable convergence factor for split-path adaptation in transform domain.
4. To investigate the innate characteristics of split operation for data compression and to compare with the transform adaptation, and to establish a unification with the discrete Walsh transform adaptation.

The thesis is organized as follows. In Chapter 2, a brief review of the adaptive signal processing and previous work on split-path adaptive filtering is given, and a split-path median LMS algorithm is proposed. Chapter 3 discusses how to construct a continuous split-path adaptive filter for general cases and a detail derivation procedure is included. In Chapter 4, the properties and performances of the transform domain adaptive algorithm for split-path structure and non-split structure are studied. The performances is also compared with their counterpart in time domain. A pair of variable convergence factors for the split structure are proposed in Chapter 5, and a optimal convergence factor tracking method is deduced for practical implementation. In Chapter 6, the inter-relationship of split-path adaptive filtering and DWT adaptation is analyzed, and a unification between them is then established. In addition, a new definition of fast Walsh transform function is also given which renders an efficient implementation scheme.

Chapter 2 TIME DOMAIN SPLIT-PATH ADAPTIVE SYSTEM

Adaptive filtering provides a powerful tool for many signal processing applications in which the input signal statistics are unknown a priori and/or the environments are slowly time variant [Sibul 1987] [Orfanidis 1988]. The most commonly used adaptive algorithm is Widrow's least mean square (LMS) algorithm [Widrow 1985]. This algorithm has received considerable attention by many researchers over the past two decades. The LMS is a relatively simple algorithm to implement that enables many real-time applications to become possible. Its properties such as steady state misadjustment [Widrow 1976] and tracking performance [Sibul 1987] have also been investigated thoroughly. Two major drawbacks of the LMS algorithm are its relatively slow convergence speed and the noisy gradient vector estimate that causes random fluctuations [Widrow 1985]. Many efforts have been made to improve the adaptation performance of the LMS algorithm particularly to speed up the convergence rate.

In this chapter, we first study the behaviour of stochastic gradient-based adaptive algorithms, specifically the properties and limitations of the LMS algorithm. Secondly, we introduce the idea of split-path adaptive filtering and then propose a split-path median LMS algorithm [Wan 1994b] [Wan 1994c] by in-corporating the split structure in median filter operation. The performance of the algorithm will be analyzed and evaluated in an application of line enhancement.

2.1 Adaptive Transversal Filter and the LMS Algorithm

2.1.1 Wiener-Hopf Solution

The finite impulse response (FIR) filter is by far the most practical and widely used filter structure for adaptive filtering. Adaptive transversal filters have a wide range of applications in communications, system modeling and system identification. The general form of an adaptive transversal filter of order (M-1) is shown in Fig. 2.1. The function of the FIR filter is to weigh and sum the tapped delayed signals $x(k)$, $x(k-1)$, ..., $x(k-M+1)$ to form an adaptive output. The filter weights or coefficients can be written in vector form,

$$\boldsymbol{w}_k = [w_0(k),\ w_1(k),\ \cdots,\ w_{M-1}(k)]^T \tag{2.1a}$$

where the superscript T denotes a transpose operation, while the tapped delay line (TDL) input vector can be expressed as

$$\boldsymbol{x}_k = [x(k),\ x(k-1),\ \cdots,\ x(k-M+1)]^T \tag{2.1b}$$

The actual response, $y(k)$, of the system is given by the inner product of $\boldsymbol{w}_k$ and $\boldsymbol{x}_k$, i.e.

$$\begin{aligned} y(k) &= w_0(k)x(k) + w_1(k)x(k-1) + \cdots + w_{M-1}(k)x(k-M+1) \\ &= \boldsymbol{w}_k^T \boldsymbol{x}_k \end{aligned} \tag{2.2}$$

Now suppose the desired response of the system is $d(k)$, then the error signal, $e(k)$, is given by the difference between $d(k)$ and $y(k)$,

$$e(k) = d(k) - y(k) = d(k) - \boldsymbol{w}_k^T \boldsymbol{x}_k \tag{2.3}$$

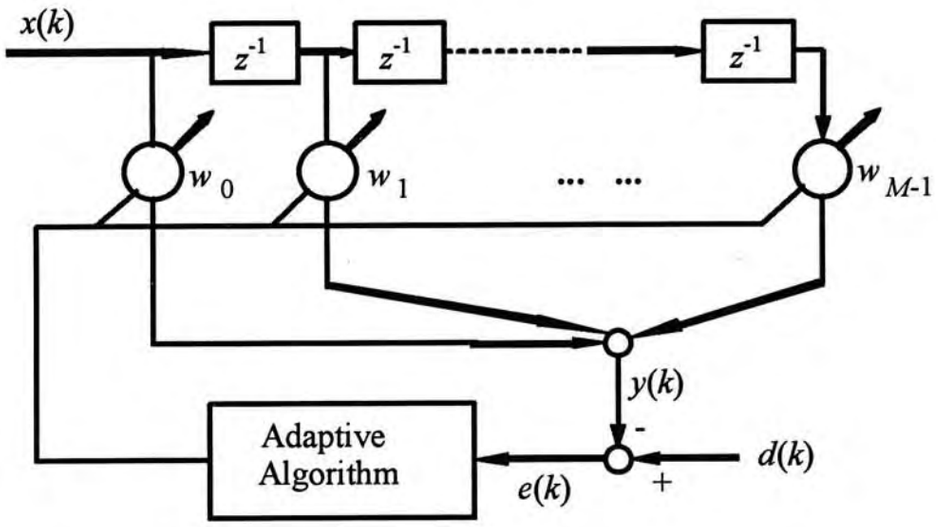


Fig. 2.1. An adaptive transversal filter.

The basic principle of operation of the adaptive system is to minimize the mean-square-error (MSE) which is defined as

$$\text{MSE} = \varepsilon \triangleq E[e^2(k)] \tag{2.4}$$

The minimization is achieved by continuously adjusting the system parameter vector $\boldsymbol{w}_k$ according to a stochastic gradient-based searching approach [Widrow 1985]. The adaptive algorithm that recursively computes the filter coefficients, and thus searches for the minimum MSE, has the form

$$\boldsymbol{w}_{k+1} = \boldsymbol{w}_k - \mu(\nabla\varepsilon)_k \tag{2.5}$$

where

$$(\nabla\varepsilon)_k = \left[\frac{\partial\varepsilon}{\partial w_0} \quad \frac{\partial\varepsilon}{\partial w_1} \quad \cdots \quad \frac{\partial\varepsilon}{\partial w_{M-1}}\right]^T \tag{2.6}$$

is the gradient vector at the kth iteration, and μ is a convergence factor or step size that controls stability and rate of convergence. When the input signals and the desired response are statistically stationary, the MSE performance surface is a quadratic function of the filter weights which can be expressed as

$$\varepsilon = E[e^2(k)] = E[d^2(k)] - 2\boldsymbol{R}_{dx}^T\boldsymbol{w}_k + \boldsymbol{w}_k^T\boldsymbol{R}_x\boldsymbol{w}_k \tag{2.7}$$

where

$$\boldsymbol{R}_x \triangleq E[\boldsymbol{x}_k\boldsymbol{x}_k^T] \tag{2.8}$$

is defined as the input autocorrelation matrix, and

$$\boldsymbol{R}_{dx} \triangleq E[d(k)\boldsymbol{x}_k] \tag{2.9}$$

is defined as the cross correlation vector, respectively. By setting the gradient of the MSE function to zero yields

$$(\nabla\varepsilon)_k = -2\boldsymbol{R}_{dx} + 2\boldsymbol{R}_x\boldsymbol{w}_k = 0 \tag{2.10}$$

The solution of (2.10) gives the optimal Wiener-Hopf weight vector,

$$\boldsymbol{w}^* = \boldsymbol{R}_x^{-1}\boldsymbol{R}_{dx} \tag{2.11}$$

and the minimum MSE of the system can be computed from

$$\varepsilon_{\min} = E[d^2(k)] - 2\boldsymbol{R}_{dx}^T\boldsymbol{w}^* \tag{2.12}$$

This is an ideal solution for the filter coefficients on the condition that a *prior knowledge* of the input characteristics and the desired response are available. However, such information only exists for a determined system, and is usually unknown in most real time applications especially when the environment is changing from time to time.

2.1.2 The LMS Adaptive Algorithm

The LMS adaptive algorithm which was first proposed by Widrow [Widrow 1960] [Koford 1966] [Widrow 1967] [Widrow 1971] is an efficient and effective method to approximate Wiener-Hopf solution on a sample-by-sample basis. By replacing the MSE with an instantaneous value, $e^2(k)$, the estimation of the gradient is simplified to

$$(\hat{\nabla}\varepsilon)_k = \nabla[e^2(k)] = -2e(k)\boldsymbol{x}_k \tag{2.13}$$

As a result, the updating equation for the weight vector becomes

$$\boldsymbol{w}_{k+1} = \boldsymbol{w}_k + 2\mu e(k)\boldsymbol{x}_k \tag{2.14}$$

This is the famous Widrow-Hoff LMS algorithm. The LMS algorithm is in fact an implementation of the steepest-descent method. Although it makes use of the gradients of the MSE function, it does not require off-line gradient estimations or

repetitions of data. From (2.14), it can be seen that the algorithm is simple and easy to implement. It does not need explicit measurements of correlation functions, nor does it involve matrix inversion. The method is indeed elegant in its simplicity and efficiency. But accuracy is limited by statistical sample size, since the filter coefficients found are based on real-time acquisition of input signals [Widrow 1975].

The LMS algorithm permits the adjustment of gain and phase at many frequencies simultaneously and is useful in adaptive broad-band signal processing. A set of simplified design rules has been given in [Widrow 1975]: The tap spacing time must be at least as short as the reciprocal of twice the signal bandwidth. The total real-time length of delay line is determined by the reciprocal of the desired filter frequency resolution. Thus, the number of weights required is generally equal to twice the ratio of the total signal bandwidth to the frequency resolution of the filter.

Since the mean value of $(\hat{\nabla}\varepsilon)_k$ is equal to the true gradient $(\nabla\varepsilon)_k$, the gradient estimate used in the LMS algorithm is unbiased and the expected value of the weight vector converges to the Wiener weight vector $\boldsymbol{w}^*$ when the input vectors are uncorrelated over time [Riegler 1973]. Starting with an arbitrary initial weight vector, the algorithm will converge in the mean and will remain stable upon the necessary condition that parameter μ satisfies [Widrow 1976]

$$0 < \mu < \frac{1}{\lambda_{\max}} \tag{2.15}$$

where $\lambda_{\max}$ is the largest eigenvalue of the matrix $\boldsymbol{R}_x$. Within these bounds, the speed of adaptation and also the noise in the weight vector solution are in general determined by the value of μ. Let λ_i to be the ith eigenvalue of $\boldsymbol{R}_x$, then the mean of $\boldsymbol{w}_k$ will

converge exponentially to $\boldsymbol{w}^*$ with the time constant of the ith natural mode given by [Widrow 1976]

$$\tau_i = \frac{1}{2\mu\lambda_i} \tag{2.16}$$

When steady state is reached, the variance of the weights due to gradient noise is given by

$$cov\{\boldsymbol{w}_k\} = \mu\,\varepsilon_{min}\boldsymbol{I}_M \tag{2.17}$$

where $\boldsymbol{I}_M$ is an identity matrix of size M. The output MSE equals

$$\varepsilon = \varepsilon_{min} + \mu\,\varepsilon_{min} tr(\boldsymbol{R}_x) \tag{2.18}$$

where ε_{min} is the minimum possible MSE, and tr($\boldsymbol{R}_x$) is the trace of the matrix $\boldsymbol{R}_x$.

During adaptation, the error $e(k)$ is nonstationary as the weight vector approaches to $\boldsymbol{w}^*$. The learning curve, which plots the MSE versus number of adaptation, is an ensemble of M exponentials and each of which corresponds to a natural mode of the algorithm. For the ith natural mode, the convergence behaviour is characterized by their individual time constant of geometric decay and can be expressed as

$$\tau_{i,\mathrm{mse}} = \frac{1}{4\mu\lambda_i} \tag{2.19}$$

The smaller the time constant, the convergence speed is faster. Obviously, the largest time constant corresponds to the slowest converging mode, i.e.

$$\tau_{\mathrm{max,mse}} = \frac{1}{4\mu\lambda_{\mathrm{min}}} \tag{2.20}$$

where λ_{min} is the minimum eigenvalue of the input correlation matrix. Combining (2.15) with (2.20), we have

$$\tau_{max,mse} > \frac{1}{4}\frac{\lambda_{max}}{\lambda_{min}} \tag{2.21}$$

The ratio of the maximum to minimum eigenvalues,

$$\gamma \triangleq \frac{\lambda_{max}}{\lambda_{min}} \tag{2.22}$$

is defined as *eigenvalue spread* of the input autocorrelation matrix. Expression (2.21) reveals an important relationship between the convergence behaviour of the LMS and the inherent characteristic of the input signal: the convergence speed will decrease as the eigenvalue spread of the input autocorrelation matrix increases. That implies the convergence rate will be relatively slow for a highly correlated input signal. This is one of the main disadvantages of the LMS algorithm.

The assumption of decorrelation and stationarity of the input vector are not necessary conditions for convergence of the LMS algorithm but have been adopted for analytic convenience. For a slowly changing nonstationary input or signal that is not total uncorrelated, the convergence can also be ensured [Widrow 1976] [Farden 1981].

2.2 Split Structure Adaptive Filtering

Many efforts have been made to improve the convergence behaviour especially to speed up the convergence rate of the adaptive algorithm. Some researchers attempt to improve the inherent characteristics of the input data to be processed, in the case of the LMS algorithm is to reduce the eigenvalue spread of the input autocorrelation matrix, such as block implementation of adaptive digital filters [Clark 1981], block

realization of multirate adaptive filters [Lee 1986a], nonlinear quantization effects in the LMS and block LMS adaptive algorithms [Bershad 1989] *et al.* On the other hand, emphasis is also put on designing new adaptive filter structure in order to cope with different kinds of signal and/or to reduce computational complexity. They include optimality in the choice of convergence factor for gradient-based adaptive algorithms [Yassa 1987], optimum gain parameter in LMS adaptation [Bershad 1987], variable step size [Harris 1986] [Kwong 1992], structural subband decomposition of FIR adaptive filters [Petraglia 1993] [Mitra 1993] [Mahalanobis 1993] and split-path structure adaptive algorithms.

In this section, we shall briefly review the structure of a split-path adaptive algorithm first presented by Ho and Ching [Ching 1991] [Ho 1991]. We shall introduce two possible realizations for the split-path adaptive filter. An application of the split structure to adaptive median filtering will be presented and analyzed in the next section.

2.2.1 Split Structure of an Adaptive Filter

Recently, split-path adaptive filtering has been demonstrated to be an effective technique in achieving a lower eigenvalue spread and a faster convergence speed [Ho 1992]. The basic idea of this method is to divide the original transversal filter into two linear phase subfilters connected in parallel, one with antisymmetric and another with symmetric property. This splitting process will also partition the eigenvalues of the input correlation matrix into two sets, giving rise to two eigenvalue spreads that are

normally smaller as compared with the non-split model. When appropriate convergence factors for the two filters are chosen in accordance with their reduced eigenvalue spreads, the system performance, specifically the convergence speed, can be fastened. The split structure adaptive filter has been applied to AR modeling [Ho 1992a], median filtering [Wan 1994c], system identification [Ho 1991a] and linear prediction [Wan 1992] with satisfactory results.

2.2.2 Split-Path Structure for a Non-Symmetric Adaptive Filter

Consider an M-coefficient non-symmetric transversal adaptive filter which is described by the following transfer function,

$$W(z) = \sum_{i=0}^{M-1} w_i z^{-i} \tag{2.23}$$

Assuming M is even and $N=M/2$, the original filter can be split into a pair of linear phase subfilters, $P(z)$ and $Q(z)$, connected in parallel, one with antisymmetric property and another with symmetric property. Their transfer functions are of the form,

$$P(z) = \sum_{i=0}^{N-1} p_i (z^{-i} - z^{-M+1+i}) \tag{2.24}$$

and

$$Q(z) = \sum_{i=0}^{N-1} q_i (z^{-i} + z^{-M+1+i}) \tag{2.25}$$

Although $P(z)$ and $Q(z)$ are both M-length filters, there are actually only N distinct parameters for each of them. Comparing (2.23) with (2.24) and (2.25), the parameters of the split filters and the parent filter are related by

$$w_i(k) = \begin{cases} p_i(k)+q_i(k) & i=0,\ 1,\ \ldots,\ N\text{-}1 \\ -p_{M-1-i}(k)+q_{M-1-i}(k) & i=N,\ N+1,\ \ldots,\ M\text{-}1 \end{cases} \tag{2.26}$$

The z-transform of the system output can be expressed as

$$Y(z) = W(z)X(z) = [P(z)+Q(z)]\,X(z) \tag{2.27}$$

where $X(z)$ is the z-transform of the input $x(k)$. In the time domain, the output of the split system at instant k is given by,

$$\begin{aligned} y(k) = & \sum_{i=0}^{N-1} p_i(k)[x(k-i)-x(k-M+1+i)] \\ & + \sum_{i=0}^{N-1} q_i(k)[x(k-i)+x(k-M+1+i)] \\ & = \sum_{i=0}^{N-1} p_i(k)s_{p,i}(k) + \sum_{i=0}^{N-1} q_i(\mathrm{k})s_{q,i}(k) \end{aligned} \tag{2.28}$$

where

$$s_{p,i}(k) = x(k-i)-x(k-M+1+i) \qquad i=0,\ 1,\ \cdots,\ N-1 \tag{2.29a}$$

and

$$s_{q,i}(k) = x(k-i)+x(k-M+1+i) \qquad i=0,\ 1,\ \cdots,\ N-1 \tag{2.29b}$$

can be considered as two parallel input sets for two linear combiner $P(z)$ and $Q(z)$, respectively. The above splitting operation can be described more succinctly by using matrix notation. Let us define,

$$\boldsymbol{U}_1 \triangleq \begin{bmatrix} \boldsymbol{I}_1 & -\boldsymbol{J}_1 \\ \boldsymbol{I}_1 & \boldsymbol{J}_1 \end{bmatrix}_{M\times M} \tag{2.30}$$

where $\boldsymbol{I}_1$ is an $N\times N$ identity matrix and $\boldsymbol{J}_1$ is an anti-diagonal identity matrix of the same size, that is,

$$\boldsymbol{J}_1 = \begin{bmatrix} 0 & \cdots & 1 \\ \vdots & \iddots & \vdots \\ 1 & \cdots & 0 \end{bmatrix}_{N\times N} \tag{2.31}$$

We call $\boldsymbol{U}_1$ an orthogonal transform matrix or "splitting matrix" because it is used to decompose the parameter vector as well as the input data into smaller parts. Let

$$\boldsymbol{p}_k = [p_0(k), p_1(k), \cdots, p_{N-1}(k)]^T \tag{2.32}$$

and

$$\boldsymbol{q}_k = [q_0(k), q_1(k), \cdots, q_{N-1}(k)]^T \tag{2.33}$$

represent the parameter vector of the two branches respectively. The new input sequences, $\boldsymbol{s}_{p,k}$ and $\boldsymbol{s}_{q,k}$, for the two branches of the split system can be obtained from

$$\begin{bmatrix} \boldsymbol{s}_{p,k} \\ \boldsymbol{s}_{q,k} \end{bmatrix} = \boldsymbol{U}_1 \boldsymbol{x}_k \tag{2.34}$$

where

$$\boldsymbol{s}_{p,k} = [s_{p,0}(k), s_{p,1}(k), \cdots, s_{p,N-1}(k)]^T \tag{2.35}$$

and

$$\boldsymbol{s}_{q,k} = [s_{q,0}(k), s_{q,1}(k), \cdots, s_{q,N-1}(k)]^T \tag{2.36}$$

Then (2.28) becomes

$$y(k) = y_p(k) + y_q(k) = \boldsymbol{s}_{p,k}^T \boldsymbol{p}_k + \boldsymbol{s}_{q,k}^T \boldsymbol{q}_k \tag{2.37}$$

where $y_p(k)$ and $y_q(k)$ are the outputs of branch $\boldsymbol{p}$ and $\boldsymbol{q}$, respectively. The relationship of the filter parameters in (2.26) can also be expressed in terms of $\boldsymbol{U}_1$, by

$$\boldsymbol{w}_k = \boldsymbol{U}_1^T \begin{bmatrix} \boldsymbol{p}_k \\ \boldsymbol{q}_k \end{bmatrix} \tag{2.38}$$

Now we have decomposed the original filter into two independent subfilters. When LMS algorithm is employed, the new parameter vector updating equations of the respective branches are

$$\boldsymbol{p}_{k+1} = \boldsymbol{p}_k + 2\mu_p\, e(k)\boldsymbol{s}_{p,k} \tag{2.39}$$

and

$$\boldsymbol{q}_{k+1} = \boldsymbol{q}_k + 2\mu_q\, e(k)\boldsymbol{s}_{q,k} \tag{2.40}$$

Then the necessary stable condition for these subsystems become

$$\mu_p < \frac{1}{\lambda_{p,max}} \quad \text{and} \quad \mu_q < \frac{1}{\lambda_{q,max}} \tag{2.41}$$

where $\lambda_{p,max}$ and $\lambda_{q,max}$ are the maximum eigenvalues of $\boldsymbol{R}_p = \mathrm{E}[\boldsymbol{s}_{p,k}\, \boldsymbol{s}^T_{p,k}]$ and $\boldsymbol{R}_q = \mathrm{E}[\boldsymbol{s}_{q,k}\, \boldsymbol{s}^T_{q,k}]$ respectively. Since the two symmetric/antisymmetric subfilters are now completely decoupled, adaptation of individual filter path can be performed independently. Furthermore, as the eigenvalue spreads of the associated input covariance matrices of the two subfilters are getting smaller, different value of convergence factors, μ_p and μ_q, can be employed for individual path. Therefore, the performance behaviour will be greatly improved.

The split structure adaptive filtering can be perceived in two different ways. The first one is to form a pair of antisymmetric/symmetric M-length filters from the parent filter and the original input $x(k)$ is then fed directly to both subfilters. The realization scheme is shown in Fig. 2.2. Alternatively, the splitting procedure can be considered as the reconstruction of two input sequences $\boldsymbol{s}_{p,k}$ and $\boldsymbol{s}_{q,k}$, from the original input by making use of the split matrix $\boldsymbol{U}_1$. These transformed signals are then applied to the respective subfilters which are both $M/2$-length with the parameter vector

represented by p_k and q_k respectively. The realization of second approach is depicted in Fig. 2.3 diagrametically. In this realization scheme, the antisymmetric/symmetric property of the subfilters has been embedded in the signal reconstruction process. Apparently, Fig. 2.2 gives a redundant realization scheme because twice the number of parameters are needed although intuitively it is easier to understand.

2.3 Split-Path Adaptive Median Filtering

In this section, application of the split-path adaptive algorithm will be studied. We shall first investigate the properties of a median filter, especially its capabilities of suppressing impulsive interferences and preserving desired edges of the signal. Then it will be shown that by applying the split structure to an adaptive line enhancer, the advantages of an adaptive median filter can be further enhanced.

2.3.1 Median Filtering and Median LMS Algorithm

The LMS method is widely used in adaptive signal processing mainly because of its simplicity and robustness to changing environments. But when the input or desired signal is corrupted by impulsive interference, that is, severe distortions appear in the gradient estimate term $e(k)\mathbf{x}_k$, the LMS algorithm will suffer serious performance degradation and may result in a loss of stability. To circumvent this problem, an averaged least mean square (ALMS) algorithm [Kim 1975] may be used to suppress such interference. The ALMS algorithm is defined as

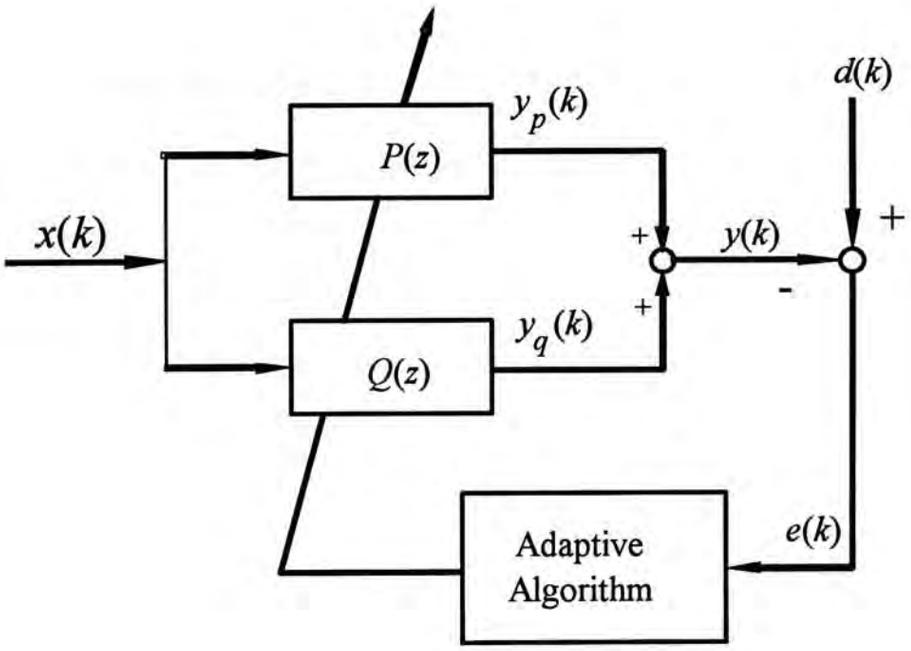


Fig. 2.2 Direct realization of the split adaptive structure.

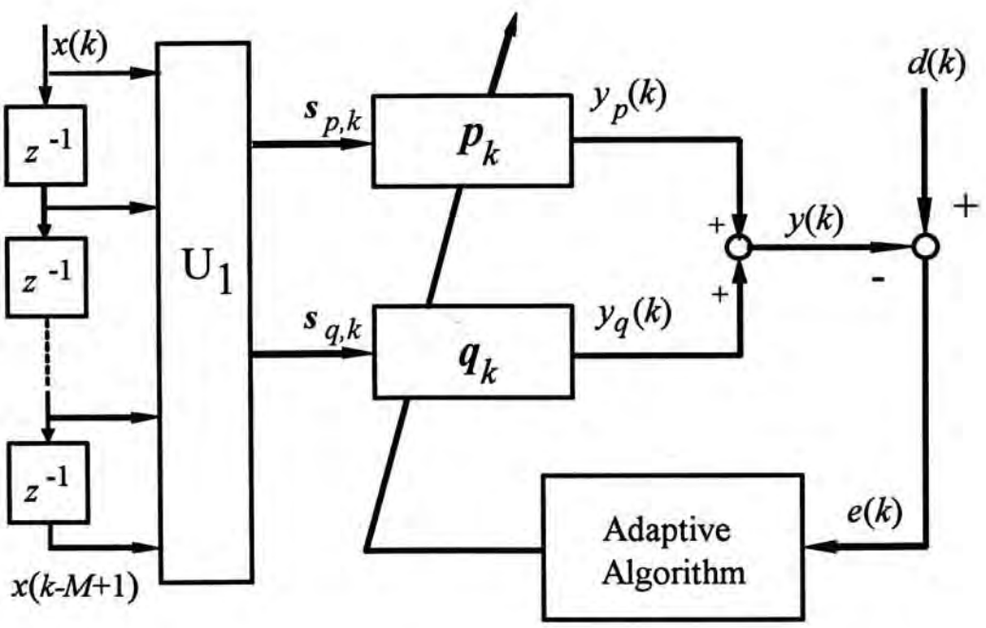


Fig. 2.3 Realization of the split adaptive filter by reconstructing new input signal.

$$\boldsymbol{w}_{k+1} = \boldsymbol{w}_k + \mu \frac{1}{N} \sum_{i=k-N+1}^{k} e(i)\boldsymbol{x}_i \tag{2.42}$$

where N signifies the averaging interval. While ALMS can reduce the level of impulse interference, it will slow down the convergence rate of the adaptation. Furthermore, it will also smooth and deteriorate the edges of the desired signal, in other words, it will reduce the capability of tracking nonstationary inputs.

The introduction of order statistic (OS), and particularly the median filter [Tukey 1977] may be one of the possible solutions. The merit of median filtering is its capability of preserving discontinuities of sufficient duration such as sharp edges or the ramps of signals while eliminating sparse impulses and local roughness such as "pepper and salt" noise from data [Huang 1981]. A median filtering operation can be expressed as

$$y(k) = \text{med}\{x_k\}_L \triangleq \text{median}\{x_{k-(L-1)/2}, \cdots, x_k, \cdots, x_{k+(L-1)/2}\} \tag{2.43}$$

where sequence $\{ x_i \}$ ranked from smallest to largest in a moving window of size L, that is, x_k is a median value of the sequence. The window is usually considered as symmetric about its centre and when window scans through the data, the output value is replaced by the median value. It is easy to see that the median filter can preserve edges, while moving average filter

$$y(k) = (x_{k-(L-1)/2} + \cdots + x_k + \cdots + x_{k+(L-1)/2}) / L \tag{2.44}$$

will change the sharp edge to L-length ramp. Therefore, the ALMS described in (2.42) is not feasible for edge preserving application.

This distinct property has led to median filter being widely used in image processing [Pratt 1978] [Restrepo 1988] and in speech processing [Rabiner 1975]. Recently, Clarkson and Haweel proposed an adaptive median LMS (MLMS) algorithm [Clarkson 1989] to facilitate performance gains over a wide range of input data types corrupted by impulsive interference.

Although the MLMS algorithm has been shown efficaciously in suppressing sparse impulses embedded in the original signal [Haweel 1992], its relatively slow convergence rate is still not better than the conventional LMS algorithm. This obstacle prevents the MLMS algorithm to be used efficiently to track the sharp edges of the desired signal in real-time applications. To overcome this difficulty, a fast response non-linear adaptive filtering method is proposed which can improve the convergence speed of order statistics LMS systems. The algorithm makes use of a split adaptive filter structure and the median LMS technique and is, therefore, called the split-path median LMS (SPMLMS) algorithm [Wan 1994b]. The algorithm is applied for line enhancement in a noisy environment. It is demonstrated that the proposed method can suppress impulse interference effectively and preserve edge details of the signal while significantly improve the convergence characteristics of the adaptive process [Wan 1994c].

2.3.2 The Split-Path Median LMS (SPMLMS) Algorithm

Consider a simple edge-extraction problem where the input sequence $x(k)$ consists of a step input $h(k)$ with signal level c and contaminated by a Gaussian white

noise $r(k)$. In this case, $x(k)$ is independent identically distributed (i.i.d.) with symmetric density and $E[x(k)] = 0$ at one side of the edge, whilst $E[x(k)] = c$ at the other side of the step function. Fig. 2.4 shows the block diagram of an adaptive line enhancer, where the delay Δ is introduced to decorrelate the broadband noise between $x(k)$ and $x(k-\Delta)$. Assume the order of an adaptive FIR filter, M, to be even and $M=2N$, where N is a positive integer. By employing the splitting technique, the conventional transversal filter can be split into a pair of subfilters connected in parallel. One of these two filters has antisymmetric property while the other has symmetric property, and their transfer functions can be described by (2.24) and (2.25), respectively.

The input vectors, $\boldsymbol{s}_{p,k}$ and $\boldsymbol{s}_{q,k}$, of the two split paths are now both of order N and can be obtained from

$$\begin{bmatrix} \boldsymbol{s}_{p,k} \\ \boldsymbol{s}_{q,k} \end{bmatrix} = \begin{bmatrix} \boldsymbol{I}_N & -\boldsymbol{J}_N \\ \boldsymbol{I}_N & \boldsymbol{J}_N \end{bmatrix} \boldsymbol{x}_k \tag{2.45}$$

where $\boldsymbol{I}_N$ is an identity matrix and $\boldsymbol{J}_N$ is an anti-diagonal identity matrix and both of them are of rank N as defined in (2.31). The system output, $y(k)$, is the sum of the outputs of the two filter branches and is given by

$$y(k)=\frac{1}{2}[y_p(k)+y_q(k)] \tag{2.46}$$

where

$$y_p(k) = \boldsymbol{p}_k^T \boldsymbol{s}_{p,k-\Delta} \quad \text{and} \quad y_q(k) = \boldsymbol{q}_k^T \boldsymbol{s}_{q,k-\Delta} \tag{2.47}$$

The overall system error, $e(k)$, is the averages of those appeared in the two branches,

$$e(k)=\frac{1}{2}[e_p(k)+e_q(k)] \tag{2.48}$$

where,

$$e_p(k) = [x(k) - x(k-M-1)] - y_p(k) = [x(k) - x(k-M-1)] - \boldsymbol{p}_k^T \boldsymbol{s}_{p,k-\Delta} \quad (4.49)$$

and

$$e_q(k) = [x(k) + x(k-M-1)] - y_q(k) = [x(k) + x(k-M-1)] - \boldsymbol{q}_k^T \boldsymbol{s}_{q,k-\Delta} \quad (4.50)$$

By applying LMS adaptation independently to the two paths but using the median of the gradient estimates from a moving window instead of the instantaneous values, we obtain the split-path median least-mean-square (SPMLMS) algorithm. The block diagram of the split-path adaptive median LMS line enhancement is shown in Fig. 2.5. The weight vectors $\boldsymbol{p}_k$ and $\boldsymbol{q}_k$ are now updated by the following equations,

$$\boldsymbol{p}_{k+1} = \boldsymbol{p}_k + \mu_p \text{med}\{e_p(k)\boldsymbol{s}_{p,k}\}_L \quad (2.51)$$

and

$$\boldsymbol{q}_{k+1} = \boldsymbol{q}_k + \mu_q \text{med}\{e_q(k)\boldsymbol{s}_{q,k}\}_L \quad (2.52)$$

where μ_p and μ_q are step sizes that control convergence speed of the adaptation. Notation $\text{med}\{.\}_L$ denotes the median operator of a sliding window spanning L sample points of the gradient estimates which are given by

$$\{ e_p(k)\boldsymbol{s}_{p,k},\ e_p(k-1)\boldsymbol{s}_{p,k-1},\ \cdots,\ e_p(k-L+1)\boldsymbol{s}_{p,k-L+1} \} \quad (2.53)$$

and

$$\{ e_q(k)\boldsymbol{s}_{q,k},\ e_q(k-1)\boldsymbol{s}_{q,k-1},\ \cdots,\ e_q(k-L+1)\boldsymbol{s}_{q,k-L+1} \} \quad (2.54)$$

respectively. The original median filter is now composed of two MLMS adaptive branches.

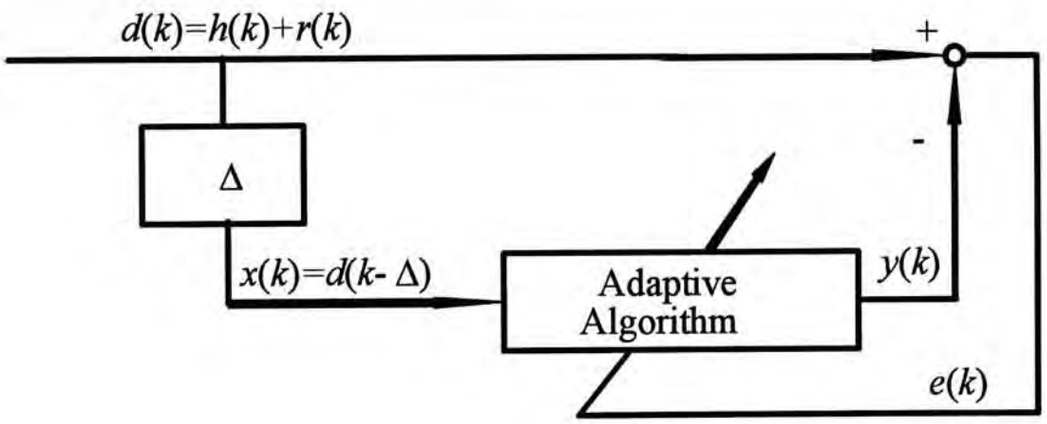


Fig. 2.4. A typical adaptive line enhancer

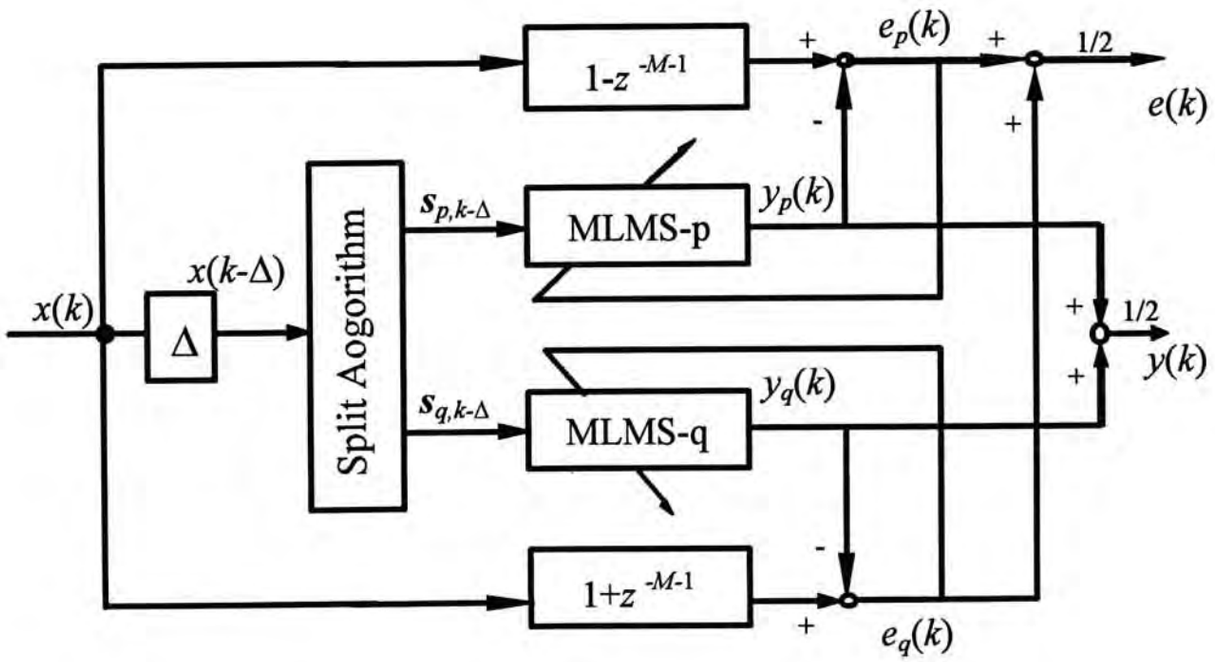


Fig. 2.5 An adaptive split-path median LMS line enhancer.

Next, we shall prove that $e_p(k)$ and $e_q(k)$ are uncorrelated to each other. For example, when $\Delta = 1$, we have

$$\begin{aligned} e_p(k) &= [x(k) - x(k-M-1)] - \boldsymbol{p}_k^T \begin{bmatrix} \boldsymbol{I}_N & -\boldsymbol{J}_N \end{bmatrix} \boldsymbol{x}_{k-1} \\ &= \begin{bmatrix} 1 & -\boldsymbol{p}_k^T \end{bmatrix} \begin{bmatrix} \boldsymbol{I}_{N+1} & -\boldsymbol{J}_{N+1} \end{bmatrix} \boldsymbol{x}_{k+2} \end{aligned} \tag{2.55}$$

and

$$\begin{aligned} e_q(k) &= [x(k) + x(k-M-1)] - \boldsymbol{q}_k^T \begin{bmatrix} \boldsymbol{I}_N & \boldsymbol{J}_N \end{bmatrix} \boldsymbol{x}_{k-1} \\ &= \begin{bmatrix} 1 & -\boldsymbol{q}_k^T \end{bmatrix} \begin{bmatrix} \boldsymbol{I}_{N+1} & \boldsymbol{J}_{N+1} \end{bmatrix} \boldsymbol{x}_{k+2} \end{aligned} \tag{2.56}$$

where $\boldsymbol{I}_{N+1}$ and $\boldsymbol{J}_{N+1}$ are matrices of rank $N+1$. Hence, we have,

$$\begin{aligned} E[e_p(k)e_q(k)] &= \begin{bmatrix} 1 & -\boldsymbol{p}_k^T \end{bmatrix} \begin{bmatrix} \boldsymbol{I}_{N+1} & -\boldsymbol{J}_{N+1} \end{bmatrix} E[\boldsymbol{x}_{k+2}\boldsymbol{x}_{k+2}^T] \begin{bmatrix} \boldsymbol{I}_{N+1} \\ \boldsymbol{J}_{N+1} \end{bmatrix} \begin{bmatrix} 1 \\ -\boldsymbol{q}_k \end{bmatrix} \\ &= \begin{bmatrix} 1 & -\boldsymbol{p}_k^T \end{bmatrix} \begin{bmatrix} \boldsymbol{I}_{N+1} & -\boldsymbol{J}_{N+1} \end{bmatrix} \boldsymbol{R}_{M+2} \begin{bmatrix} \boldsymbol{I}_{N+1} \\ \boldsymbol{J}_{N+1} \end{bmatrix} \begin{bmatrix} 1 \\ -\boldsymbol{q}_k \end{bmatrix} \\ &= \begin{bmatrix} 1 & -\boldsymbol{p}_k^T \end{bmatrix} \begin{bmatrix} \boldsymbol{I}_{N+1} & -\boldsymbol{J}_{N+1} \end{bmatrix} \begin{bmatrix} 0 & \boldsymbol{J}_{N+1} \\ \boldsymbol{J}_{N+1} & 0 \end{bmatrix} \boldsymbol{R}_{M+2} \begin{bmatrix} 0 & \boldsymbol{J}_{N+1} \\ \boldsymbol{J}_{N+1} & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{I}_{N+1} \\ \boldsymbol{J}_{N+1} \end{bmatrix} \begin{bmatrix} 1 \\ -\boldsymbol{q} \end{bmatrix} \end{aligned} \tag{2.57}$$

Use the properties of $\boldsymbol{J}$, we can get,

$$\begin{aligned} E[e_p(k)e_q(k)] &= \begin{bmatrix} 1 & -\boldsymbol{p}_k^T \end{bmatrix} \begin{bmatrix} -\boldsymbol{I}_{N+1} & \boldsymbol{J}_{N+1} \end{bmatrix} \boldsymbol{R}_{M+2} \begin{bmatrix} \boldsymbol{I}_{N+1} \\ \boldsymbol{J}_{N+1} \end{bmatrix} \begin{bmatrix} 1 \\ -\boldsymbol{q}_k \end{bmatrix} \\ &= -E[e_p(k)e_q(k)] \\ &= 0 \end{aligned} \tag{2.58}$$

This implies that minimization of individual mean square error (MSE) of the two branches separately is equivalent to minimizing the global MSE of the system.

2.3.3 Convergence Analysis of SPMLMS

Basically, only a monotone sequence will be invariant when passing through an ordinary median filter [Tukey 1977]. But this restriction can be released if the window

size of the median filter is properly chosen within a well defined range [Bovik 1983]. Given an input sequence to be median filtered with an $L=2m+1$ window, a necessary and sufficient condition for the signal to be invariant is that the transition from an ascending slope to a descending slope or *vice versa* must be separated by a constant neighborhood (i.e. at least $m+1$ consecutive identical points) or the data sequence must be *local monotone* for more than $m+2$ consecutive points [Gallagher 1981]. The commonly used square wave or a sine wave signal can be described as local monotone sequence. If $e_p(k)\boldsymbol{s}_{p,k}$ and $e_q(k)\boldsymbol{s}_{q,k}$ are decreasing in a local monotone manner, then each coefficient being updated by SPMLMS will also converge monotonically [Tukey 1977]. Because $x(k)$ is assumed i.i.d. with symmetric density and that $\boldsymbol{s}_{p,k}$ and $\boldsymbol{s}_{q,k}$ are generated by taking an orthogonal transformation of $\boldsymbol{x}_k$, the elements in $\boldsymbol{s}_p(k)$ and $\boldsymbol{s}_q(k)$ are i.i.d. In most cases, $e(k)$ is also i.i.d. with symmetric density. Therefore, the densities of the composite sequences in $e_p(k)\boldsymbol{s}_{p,k}$ and $e_p(k)\boldsymbol{s}_{q,k}$ may be considered as approximately i.i.d. and symmetric. Invoking the well-known result on the median of an i.i.d. random process [Bovik 1983], *viz*, the mean of an i.i.d. sampled signal with symmetric density is preserved under median filtering, we have

$$E\,[\mathrm{med}\{e_p(k)\boldsymbol{s}_{p,k}\}_L]=E\,[e_p(k)\boldsymbol{s}_{p,k}] \tag{2.59}$$

and

$$E\,[\mathrm{med}\{e_q(k)\boldsymbol{s}_{q,k}\}_L]=E\,[e_q(k)\boldsymbol{s}_{q,k}] \tag{2.60}$$

Taking the mean on (2.51) and using (2.59), we have

$$\begin{aligned} E[\boldsymbol{p}_{k+1}] &= E[\boldsymbol{p}_k]+\mu_p E[\mathrm{med}\{e_p(k)\boldsymbol{s}_{p,k}\}_L] \\ &= E[\boldsymbol{p}_k]+\mu_p E[e_p(k)\boldsymbol{s}_{p,k}] \\ &= E[\boldsymbol{p}_k]+\mu_p E[(d(k)-\boldsymbol{p}_k^T\boldsymbol{s}_{p,k})\boldsymbol{s}_{p,k}] \end{aligned} \tag{2.61}$$

Because the weight vector changes very slowly during adaptation, it can be taken as statistical uncorrelated with input signal, then we have

$$\begin{aligned} E[\boldsymbol{p}_{k+1}] &= E[\boldsymbol{p}_k] + \mu_p E[(d(k) - \boldsymbol{s}_{p,k}^T \boldsymbol{p}_k)\boldsymbol{s}_{p,k}] \\ &= E[\boldsymbol{p}_k] + \mu_p (\boldsymbol{R}_{dp} - \boldsymbol{R}_p E[\boldsymbol{p}_k]) \end{aligned} \tag{2.62}$$

where $\boldsymbol{R}_{dp} = \mathrm{E}[d(k)\boldsymbol{s}_{p,k}^T]$. In steady state, if the optimal solution is obtained, we have

$$\begin{aligned} E[\boldsymbol{p}_{k+1}] &= (\boldsymbol{I} - \mu_p \boldsymbol{R}_p) E[\boldsymbol{p}_k] + 2\mu_p \boldsymbol{R}_{dp} \\ &= (\boldsymbol{I} - \mu_p \boldsymbol{R}_p) E[\boldsymbol{p}_k] + 2\mu_p \boldsymbol{R}_p \boldsymbol{p}^* \end{aligned} \tag{2.63}$$

Similar result can also be derived for parameter $\boldsymbol{q}_k$. When μ_p and μ_q are selected to satisfy

$$0 < \mu_p < \frac{1}{\lambda_{p,max}} \quad \text{and} \quad 0 < \mu_q < \frac{1}{\lambda_{q,max}} \tag{2.64}$$

such that $\boldsymbol{p}_k$ and $\boldsymbol{q}_k$ will asymptotically converge to their optimal value $\boldsymbol{p}^*$ and $\boldsymbol{q}^*$. Comparing (2.64) with (2.41), we can see that convergence is also assured for the SPMLMS algorithm subject to the same stability conditions.

Although the splitting procedure does not alter the eigenvalue ratio of the global system, the eigenvalue spread of individual signal feeding to the two filter branches is always smaller than the original one [Ho 1992]. Hence an improvement in convergence behaviour is anticipated.

If the window length of the two-path median filter is short in comparison with the adaptive process and is less than twice of the identical points of the input, the monotone characteristics can be retained in the SPMLMS method. That is the edge details of the signal will usually be preserved but the sparse impulse noise interference

will be eliminated. Simulations show that when L=3 or 5, the SPMLMS algorithm can satisfactorily track square waves, periodic signals or even some pseudo periodic data like speech signals.

In many adaptive applications, $e(k)$ acts as a feedback signal to the system and is a nonstationary sequence, such that $e(k)\boldsymbol{x}_k$ is not strictly i.i.d.. Then (2.59) and (2.60) do not exit. In such cases, we can prove that the SPMLMS algorithm still converges in the mean sense under some specified conditions. Define the parameter error vectors as follows,

$$\boldsymbol{v}_{p,k} = \boldsymbol{p}_k - \boldsymbol{p}^* \quad \text{and} \quad \boldsymbol{v}_{q,k} = \boldsymbol{q}_k - \boldsymbol{q}^* \tag{2.65}$$

Substitute (2.65) into (2.51), the coefficient error vector for branch $\boldsymbol{p}$ can be expressed as

$$\begin{aligned} \boldsymbol{v}_{p,k+1} &= \boldsymbol{v}_{p,k} + \mu_p \text{med}\{(\boldsymbol{p}^{*T}\boldsymbol{s}_{p,k} - \boldsymbol{p}_k^T\boldsymbol{s}_{p,k})\boldsymbol{s}_{p,k}\}_L \\ &= \boldsymbol{v}_{p,k} - \mu_p \text{med}\{\boldsymbol{s}_{p,k}\boldsymbol{s}_{p,k}^T\boldsymbol{v}_{p,k}\}_L \end{aligned} \tag{2.66}$$

In right side of (2.66), the median gradient for updating the ith coefficient of $\boldsymbol{p}_k$ can be written in the forms of

$$\begin{aligned} &\text{med}\{s_p(k-i)s_p(k)v_{p,k}(0)+\cdots+s_p^2(k-i)v_{p,k}(i)+\cdots\}_L \\ &\quad = \text{med}\{[x(k-i)-x(k-M+1+i)][x(k)-x(k-M+1)]v_{p,k}(0) \\ &\qquad +\cdots+[x(k-i)-x(k-M+1+i)]^2 v_{p,k}(i)+\cdots\}_L \end{aligned} \tag{2.67}$$

If i < M/2, and assume that $x(k)$ is a random process such that the median filter has a "diagonal dominance" behaviour [Johnson 1988] [Haweel 1992], then taking expectation of (2.67) yields,

$$\begin{aligned}&\mathrm{E}[\mathrm{med}\{s_p(k-i)s_p(k)v_{p,k}(0)+\cdots+s_p^2(k-i)v_{p,k}(i)+\cdots\}_L]\\&\quad=\mathrm{E}[\mathrm{med}\{[x^2(k-i)+x^2(k-M+1+i)]\,v_{p,k}(i)\}_L]\end{aligned}\tag{2.68}$$

When μ_p is chosen to be small enough, the fluctuation of $\boldsymbol{v}_{p,k}$ will be very small, which can be regarded as a constant and is independent to $\boldsymbol{s}_{p,k}$. In this case,

$$\begin{aligned}&\mathrm{E}[\mathrm{med}\{[x^2(k-i)+x^2(k-M+1+i)]\,v_{p,k}(i)\}_L]\\&\quad=\mathrm{E}[\mathrm{med}\{x^2(k-i)+x^2(k-M+1+i)\}_L]\,\mathrm{E}[v_{p,k}(i)]\end{aligned}\tag{2.69}$$

Furthermore, when $\boldsymbol{x}_k$ is stationary,

$$\begin{aligned}&\mathrm{E}[\mathrm{med}\{x^2(k-i)+x^2(k-M+1+i)\}_L]\\&\quad=2\mathrm{E}[\mathrm{med}\{x^2(k-i)\}_L]\\&\quad=2\rho_p\end{aligned}\tag{2.70}$$

Thus the expectation of the parameter error vector of branch $\boldsymbol{p}$ will satisfy

$$\mathrm{E}[\boldsymbol{v}_{p,k+1}]=(1-2\mu_p\rho_p)\mathrm{E}[\boldsymbol{v}_{p,k}]\tag{2.71}$$

Similar results can be obtained for branch $\boldsymbol{q}$. Consequently the stability bounds of the step sizes becomes,

$$0<\mu_p<\frac{1}{\rho_p}\quad\text{and}\quad 0<\mu_q<\frac{1}{\rho_q}\tag{2.72}$$

The above analysis is similar to [Haweel 1992], but the result in (2.72) has indicated that the dynamic range of the convergence step sizes for the SPMLMS algorithm is approximately twice as large as that for the MLMS algorithm [Haweel 1992]. The convergence factors can be estimated from the experimental input data, or by an optimal convergence factor tracking method [Wan 1994a] [Yassa 1987]. Therefore, if μ_p and μ_q are chosen properly, a faster convergence speed can be achieved by SPMLMS.

2.4 Computer Simulation Examples

We demonstrate the superior performance of the SPMLMS algorithm in the context of line enhancement when the input signal is corrupted by noisy interferences. In our simulations, we chose the order of the filter M=8, the moving window size L=3, and the time delay Δ=7. The input signal employed was in the form of

$$x(k) = h(k) + r(k) \tag{2.73}$$

where $h(k)$ was the desired signal contaminated by another undesired noise, $r(k)$.

Fig. 2.6 plots the learning curves obtained by using different algorithms including the conventional LMS, MLMS and SPMLMS. The convergence factors were kept identical in all cases. The input signal $h(k)$, in this example, was a step function with amplitude equal to 1.5, while the undesirable signal was a Gaussian white noise with power σ^2=0.025. The convergence factors were chosen as $\mu=\mu_p=\mu_q$=0.0005. Apparently, SPMLMS performs better than the conventional LMS and MLMS. It tracks the step input with a fastest convergence rate among three methods. In Fig. 2.7, a squarewave input with varying pulse width and amplitude was used in order to evaluate the edge-tracking capability of the algorithms more rigorously. In this experiment, the noise power and the convergence factors were set as σ^2=1 and $\mu=\mu_p=\mu_q$=0.0001 respectively. The output responses of the algorithms are depicted for comparison. It is noted that SPMLMS is capable of capturing the original shape of the abruptly changing edges of the input signal in a much shorter interval and with smaller ripples.

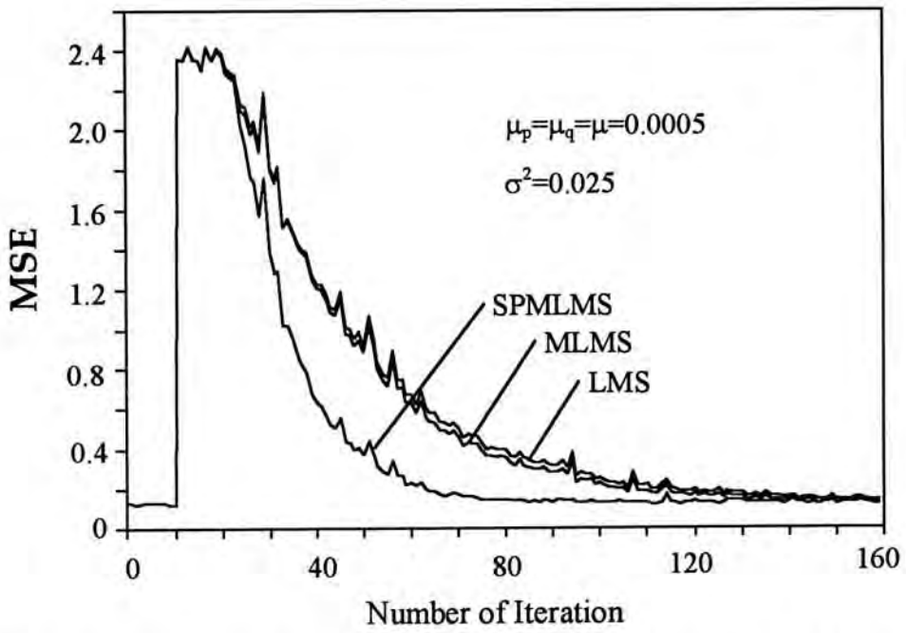


Fig. 2.6 Comparison of convergence speed for SPMLMS, MLMS and LMS.

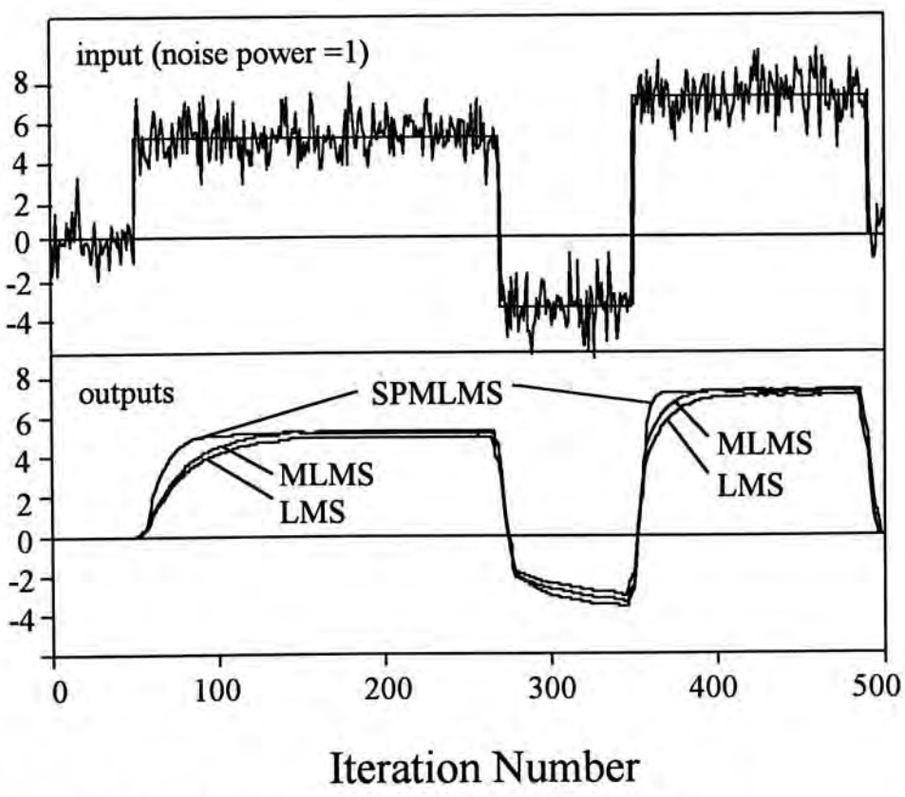


Fig. 2.7. Output responses for square wave input.

Fig. 2.8 compares the impulse rejection capabilities of LMS, MLMS and SPMLMS. In this case, two impulsive interferences of amplitude 10 were added to the desirable signal $h(k)$ which was a sinusoid in the presence of a white Gaussian noise. It is observed that both MLMS and SPMLMS suppress the impulsive interferences efficaciously while the wave distortion for LMS is most serious. In this experiment, we have used different convergence factors for different algorithms in order to produce the best results for each individual method. If we use the same small value of μ for LMS and MLMS as for SPMLMS, MLMS will converge much slower whilst the convergence rate of LMS will be so slow that the desired signal can never be extracted from the noisy environment. This demonstrates that SPMLMS has a superior performance and a faster convergence speed than other methods [Wan 1994c].

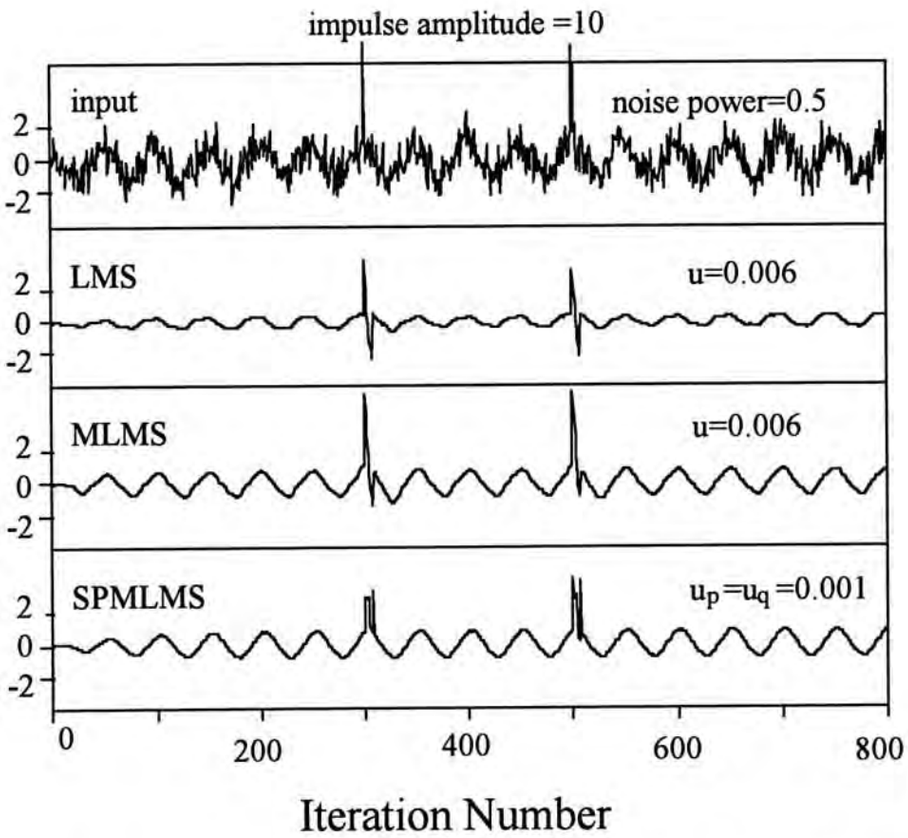


Fig. 2.8. Performance comparison of LMS, MLMS and SPMLMS to the impulsive distortion.

2.5 Summary

In this chapter, we briefly review the properties of Wiener-Hopf solution and the behaviour of stochastic gradient-based adaptive algorithms, especially the characteristics and limitation of the well-known LMS algorithm. We then introduce the principle of split-path adaptive filtering and discuss the advantages of the method. Two realization schemes for the split-path adaptive algorithm have been described.

As an application, a split-path median adaptive algorithm is proposed that makes use the merits of both the split structure and the median filtering technique. The convergence behaviour of the algorithm has been investigated, which indicates that the split-path median LMS is very effective for suppressing the sparse impulsive noise and for tracking the sharp edges of the desired information. Computer simulations of an application to line enhancement are presented to confirm the superiority of the proposed algorithm.

Chapter 3 MULTI-STAGE SPLIT STRUCTURE ADAPTIVE FILTERING

The split-path adaptive algorithm has been shown to possess many distinct advantages for adaptive signal processing in spite of the fact that it is mainly used for non-symmetric /antisymmetric systems.

In this chapter, we shall extend the split-path structure to a more general modular form, namely the multi-stage split-path adaptive filtering approach [Wan 1995a]. We shall illustrate that any systems with $M=2^L$ coefficients can be decomposed, after L steps splitting operation, into M single parameter adaptive subunits connected in parallel. The properties of the proposed algorithm will be discussed in terms of the mean square error and the convergence rate of the system. The multi-stage split-path adaptive filter has many unique merits, especially in transforming the original input data to some sequences that possess specific characteristics for easy of processing, which enables it to have superior adaptation performance than other conventional adaptive algorithms. The computation involved in the new algorithm is straight-forward and succinct, therefore, it is easy to be exploited in real time applications [Wan 1995c].

3.1 Introduction

In Chapter 2, we have shown that the split-path structure is very effective in improving the performance of an adaptive system particularly to enhance the

convergence rate of the algorithm. It is intuitive to think that if $P(z)$ or $Q(z)$ is split once again, the eigenvalues in associated branches will be redistributed into two sequences each with a smaller eigenvalue spread, consequently faster convergence behaviour could be anticipated. Furthermore, if the original FIR filter is of length $M=2^L$, the question of whether the above splitting procedure can be applied stage by stage in order to achieve an entirely parallel structure is of great interest. We shall illustrate in this section that a symmetric or an antisymmetric filter cannot be split into two linear phase subfilters by directly employing the above technique. Subsequently, we shall propose a novel method that can decompose an FIR filter step by step in a pragmatical way by using the split-path configuration.

The equation (2.38) that illustrates the relationship between the filter parameters can be expressed in the form of

$$\begin{bmatrix} \boldsymbol{p}_k \\ \boldsymbol{q}_k \end{bmatrix} = (\boldsymbol{U}_1^T)^{-1} \boldsymbol{w}_k \tag{3.1}$$

Because $\boldsymbol{U}_1$ is a "normalized orthogonal transform matrix", that is

$$\boldsymbol{U}_1^T \boldsymbol{U}_1 = 2\boldsymbol{I} \tag{3.2}$$

We have

$$(\boldsymbol{U}_1^T)^{-1} = \frac{1}{2}\boldsymbol{U}_1 \tag{3.3}$$

Assume $W(z)$ is already a symmetric filter, then after the splitting operation, (3.1) will become

$$\begin{bmatrix} \boldsymbol{p}_k \\ \boldsymbol{q}_k \end{bmatrix} = \frac{1}{2}\boldsymbol{U}_1 \boldsymbol{w}_k$$

$$= \frac{1}{2}\begin{bmatrix} \boldsymbol{I}_1 & -\boldsymbol{J}_1 \\ \boldsymbol{I}_1 & \boldsymbol{J}_1 \end{bmatrix} \begin{bmatrix} w_0(k) \\ w_1(k) \\ \vdots \\ w_1(k) \\ w_0(k) \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ w_0(k) \\ \vdots \\ w_{N-1}(k) \end{bmatrix} \tag{3.4}$$

Therefore $\boldsymbol{p}_k = 0$. In much the same way, if $W(z)$ is an antisymmetric filter, the split operation will result in $\boldsymbol{q}_k=0$. Obviously, no benefits can be obtained from the above operation because some parameters are canceled out during the process. In order to establish a multi-stage split-path adaptive filtering structure, some modifications are needed.

3.2 Split Structure for a Symmetric or an Antisymmetric Adaptive Filter

By examining the property of linear phase antisymmetric/symmetric filters carefully, we can express the transfer function of the parent filter by using the following equation,

$$\begin{aligned} W(z) &= P(z) + Q(z) \\ &= [p_0(k)z^0 + \cdots + p_{N-1}(k)z^{-N+1}] - z^{-N}[p_{N-1}(k)z^0 + \cdots + p_0(k)z^{-N+1}] \\ &\quad + [q_0(k)z^0 + \cdots + q_{N-1}(k)z^{-N+1}] + z^{-N}[q_{N-1}(k)z^0 + \cdots + q_0(k)z^{-N+1}] \\ &= P_I(z) - z^{-N}P_D(z) + Q_I(z) + z^{-N}Q_D(z) \end{aligned} \tag{3.5}$$

where $P_D(z)$ and $Q_D(z)$ are FIR filters similar to $P_I(z)$, and $Q_I(z)$, respectively, but with the filter coefficients being arranged in reverse order. Expression (3.5) can be

considered as a description of an adaptive system that contains four separated subfilters connected in parallel. This is shown diagramatically in Fig. 3.1(b). Since $P_I(z)$, $P_D(z)$, $Q_I(z)$ and $Q_D(z)$ are no longer symmetric or antisymmetric, *each of them* itself is possible to be partitioned into a pair of symmetric/antisymmetric N-length filters. Therefore, the system can be reconstructed by eight adaptive units as depicted in Fig. 3.1(c). The splitting technique can be applied to these eight subfilters again until eventually a collection of parallel subunits each of filter length of 2 is achieved.

Based on the above discussion, we shall develop our multi-stage splitting strategy in matrix form. Let us rewrite the antisymmetric subfilter $P(z)$ as follows,

$$\begin{aligned} P(z) &= \left[z^0, z^{-1}, \cdots, z^{-M+1}\right] \left[p_0, p_1, \cdots, p_{N-1}, -p_{N-1}, \cdots, -p_1, -p_0\right]^T \\ &= \left[z^0, z^{-1}, \cdots, z^{-N+1}\right] \left[p_0(k),\ p_1(k),\ \cdots,\ p_{N-1}(k)\right]^T \\ &\quad -z^{-N}\left[z^0, z^{-1}, \cdots, z^{-N+1}\right] \left[p_{N-1}(k),\ \cdots, p_1(k),\ p_0(k)\right]^T \end{aligned} \tag{3.6}$$

Notice that the two polynomials at the right hand side of (3.6) have exactly the same elements but with reverse order, therefore

$$\begin{aligned} P(z) &= P_I(z) - z^{-N} P_D(z) \\ &= \left[z^0, z^{-1}, \cdots, z^{-N+1}\right] \boldsymbol{I}_1 \boldsymbol{p}_k - z^{-N}\left[z^0, z^{-1}, \cdots, z^{-N+1}\right] \boldsymbol{J}_1 \boldsymbol{p}_k \end{aligned} \tag{3.7}$$

$P_I(z)$ and $P_D(z)$ represent two N-length filters with identical parameters but arranging in forward and backward configuration and they are neither symmetric nor antisymmetric. Therefore, each of them can be decomposed according to the procedure as described in Chapter 2. Define $\boldsymbol{I}_2$ as an identity matrix and $\boldsymbol{J}_2$ as an anti-diagonal identity matrix of rank $N/2$, $\boldsymbol{p}_k$ can be separated into two vectors by a splitting matrix,

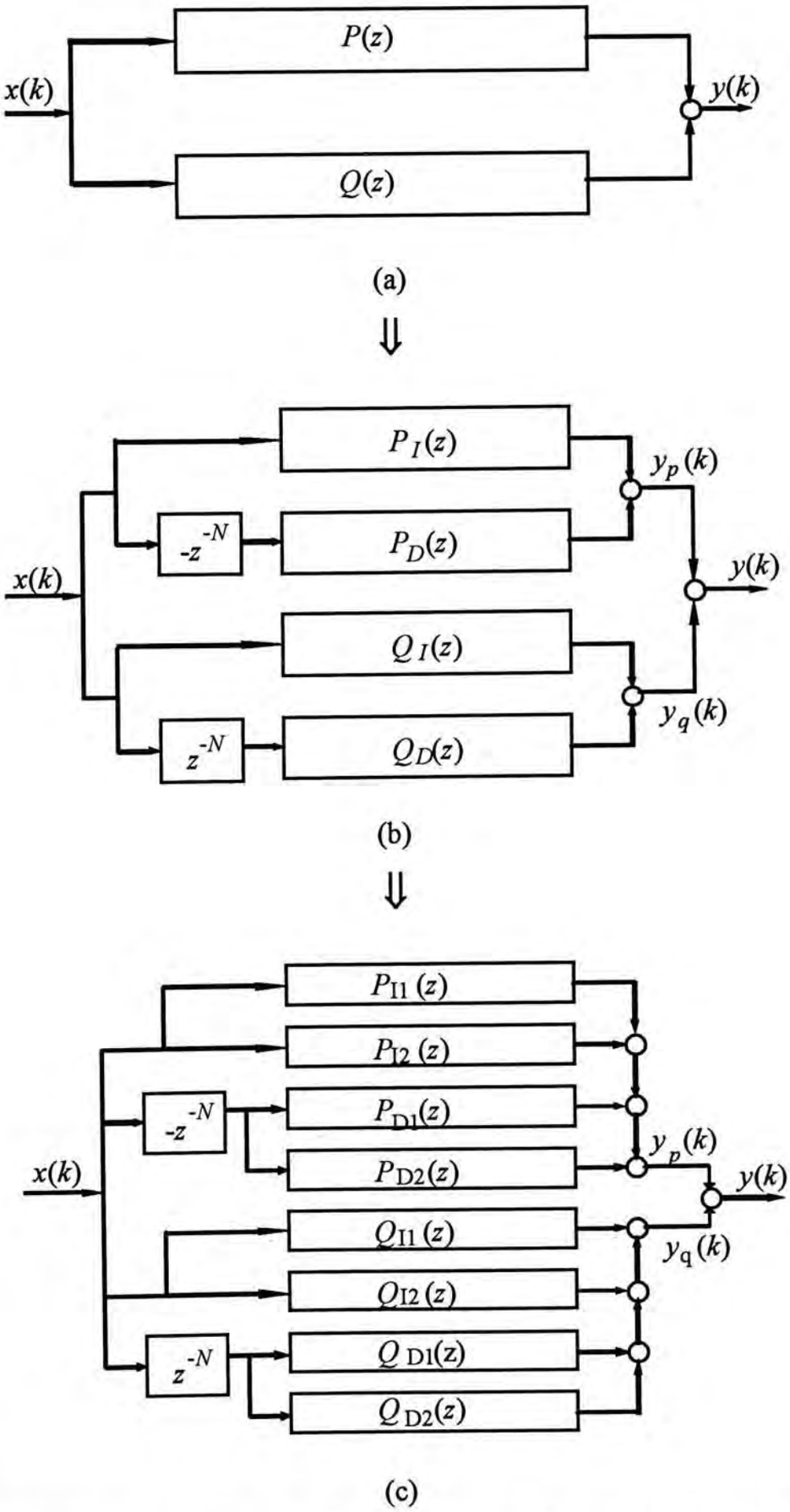


Fig. 3.1. A logical approach to split an antisymmetric filter in a modular form.

$$\boldsymbol{p}_k = \begin{bmatrix} \boldsymbol{I}_2 & \boldsymbol{I}_2 \\ -\boldsymbol{J}_2 & \boldsymbol{J}_2 \end{bmatrix} \begin{bmatrix} \boldsymbol{p}_{1k} \\ \boldsymbol{p}_{2k} \end{bmatrix} \tag{3.8}$$

where

$$\boldsymbol{p}_{1k} = [p_{1,0}(k),\, p_{1,1}(k),\, \cdots,\, p_{1,N/2-1}(k)]^T \tag{3.9}$$

and

$$\boldsymbol{p}_{2k} = [p_{2,0}(k),\, p_{2,1}(k),\, \cdots,\, p_{2,N/2-1}(k)]^T \tag{3.10}$$

Substitute $\boldsymbol{p}_k$ into (3.7), we have

$$\begin{aligned} P(z) = {} & \left[z^0, z^{-1}, \cdots, z^{-N+1}\right] \boldsymbol{I}_1 \begin{bmatrix} \boldsymbol{I}_2 & \boldsymbol{I}_2 \\ -\boldsymbol{J}_2 & \boldsymbol{J}_2 \end{bmatrix} \begin{bmatrix} \boldsymbol{p}_{1k} \\ \boldsymbol{p}_{2k} \end{bmatrix} \\ & - z^{-N} \left[z^0, z^{-1}, \cdots, z^{-N+1}\right] \boldsymbol{J}_1 \begin{bmatrix} \boldsymbol{I}_2 & \boldsymbol{I}_2 \\ -\boldsymbol{J}_2 & \boldsymbol{J}_2 \end{bmatrix} \begin{bmatrix} \boldsymbol{p}_{1k} \\ \boldsymbol{p}_{2k} \end{bmatrix} \end{aligned} \tag{3.11}$$

Now $P(z)$ is decoupled into two parts and each part is characterized by a $N/2$-length parameter vector, $\boldsymbol{p}_{1k}$ or $\boldsymbol{p}_{2k}$, accordingly. It is trivial that the above expression of the modular approach is redundant and can be simplified. By rearranging the terms in (3.11), $P(z)$ can be expressed as

$$\begin{aligned} P(z) &= \left[z^0, z^{-1}, \cdots, z^{-N+1}\right] (\boldsymbol{I}_1 - z^{-N} \boldsymbol{J}_1) \begin{bmatrix} \boldsymbol{I}_2 & \boldsymbol{I}_2 \\ -\boldsymbol{J}_2 & \boldsymbol{J}_2 \end{bmatrix} \begin{bmatrix} \boldsymbol{p}_{1k} \\ \boldsymbol{p}_{2k} \end{bmatrix} \\ &= \left[z^0, z^{-1}, \cdots, z^{-M+1}\right] \begin{bmatrix} \boldsymbol{I}_1 \\ -\boldsymbol{J}_1 \end{bmatrix} \begin{bmatrix} \boldsymbol{I}_2 & \boldsymbol{I}_2 \\ -\boldsymbol{J}_2 & \boldsymbol{J}_2 \end{bmatrix} \begin{bmatrix} \boldsymbol{p}_{1k} \\ \boldsymbol{p}_{2k} \end{bmatrix} \end{aligned} \tag{3.12}$$

The decomposition form of $Q(z)$ can be derived in a similar manner which is given by,

$$\begin{aligned} Q(z) &= \left[z^0, z^{-1}, \cdots, z^{-M+1}\right] (\boldsymbol{I}_1 + z^{-N} \boldsymbol{J}_1) \begin{bmatrix} \boldsymbol{I}_1 \\ \boldsymbol{J}_1 \end{bmatrix} \begin{bmatrix} \boldsymbol{I}_2 & \boldsymbol{I}_2 \\ -\boldsymbol{J}_2 & \boldsymbol{J}_2 \end{bmatrix} \begin{bmatrix} \boldsymbol{q}_{1k} \\ \boldsymbol{q}_{2k} \end{bmatrix} \\ &= \left[z^0, z^{-1}, \cdots, z^{-M+1}\right] \begin{bmatrix} \boldsymbol{I}_1 \\ \boldsymbol{J}_1 \end{bmatrix} \begin{bmatrix} \boldsymbol{I}_2 & \boldsymbol{I}_2 \\ -\boldsymbol{J}_2 & \boldsymbol{J}_2 \end{bmatrix} \begin{bmatrix} \boldsymbol{q}_{1k} \\ \boldsymbol{q}_{2k} \end{bmatrix} \end{aligned} \tag{3.13}$$

where

$$\boldsymbol{q}_{1k}=[q_{1,0}(k),\ q_{1,1}(k),\ \cdots,\ q_{1,N/2-1}(k)]^T \tag{3.14}$$

and

$$\boldsymbol{q}_{2k}=[q_{2,0}(k),\ q_{2,1}(k),\ \cdots,\ q_{2,N/2-1}(k)]^T \tag{3.15}$$

are the $N/2$-length parameter vectors. While the relationship

$$\boldsymbol{q}_k=\begin{bmatrix}\boldsymbol{I}_2 & \boldsymbol{I}_2\\ -\boldsymbol{J}_2 & \boldsymbol{J}_2\end{bmatrix}\begin{bmatrix}\boldsymbol{q}_{1k}\\ \boldsymbol{q}_{2k}\end{bmatrix} \tag{3.16}$$

also holds. Combining (3.12) and (3.13), the transfer function of the parent filter becomes

$$\begin{aligned}W(z)&=[z^0,z^{-1},\ \cdots,\ z^{-M+1}]\left(\begin{bmatrix}\boldsymbol{I}_1\\ -\boldsymbol{J}_1\end{bmatrix}\begin{bmatrix}\boldsymbol{I}_2 & \boldsymbol{I}_2\\ -\boldsymbol{J}_2 & \boldsymbol{J}_2\end{bmatrix}\begin{bmatrix}\boldsymbol{p}_{1k}\\ \boldsymbol{p}_{2k}\end{bmatrix}+\begin{bmatrix}\boldsymbol{I}_1\\ \boldsymbol{J}_1\end{bmatrix}\begin{bmatrix}\boldsymbol{I}_2 & \boldsymbol{I}_2\\ -\boldsymbol{J}_2 & \boldsymbol{J}_2\end{bmatrix}\begin{bmatrix}\boldsymbol{q}_{1k}\\ \boldsymbol{q}_{2k}\end{bmatrix}\right)\\ &=[z^0,z^{-1},\ \cdots,\ z^{-M+1}]\begin{bmatrix}\boldsymbol{I}_1 & \boldsymbol{I}_1\\ -\boldsymbol{J}_1 & \boldsymbol{J}_1\end{bmatrix}\begin{bmatrix}\boldsymbol{I}_2 & \boldsymbol{I}_2 & 0 & 0\\ -\boldsymbol{J}_2 & \boldsymbol{J}_2 & 0 & 0\\ 0 & 0 & \boldsymbol{I}_2 & \boldsymbol{I}_2\\ 0 & 0 & -\boldsymbol{J}_2 & \boldsymbol{J}_2\end{bmatrix}\begin{bmatrix}\boldsymbol{p}_{1k}\\ \boldsymbol{p}_{2k}\\ \boldsymbol{q}_{1k}\\ \boldsymbol{q}_{2k}\end{bmatrix}\end{aligned} \tag{3.17}$$

Now, let us define yet a second splitting matrix, $\boldsymbol{U}_2$, as follows,

$$\boldsymbol{U}_2 \triangleq \begin{bmatrix}\boldsymbol{I}_2 & -\boldsymbol{J}_2 & 0 & 0\\ \boldsymbol{I}_2 & \boldsymbol{J}_2 & 0 & 0\\ 0 & 0 & \boldsymbol{I}_2 & -\boldsymbol{J}_2\\ 0 & 0 & \boldsymbol{I}_2 & \boldsymbol{J}_2\end{bmatrix} \tag{3.18}$$

Equation (3.17) is then simplified to

$$W(z)=\left[z^{0},z^{-1},\cdots,z^{-M+1}\right]\boldsymbol{U}_1^T\boldsymbol{U}_2^T\begin{bmatrix}\boldsymbol{p}_{1k}\\ \boldsymbol{p}_{2k}\\ \boldsymbol{q}_{1k}\\ \boldsymbol{q}_{2k}\end{bmatrix} \tag{3.19}$$

This illustrates that the original adaptive filter can be split into four parallel subfilters after two splitting operations by means of $\boldsymbol{U}_1$ and $\boldsymbol{U}_2$. The total number of filters weights remains unchanged and the parameter vectors are related by

$$\boldsymbol{w}_k=\boldsymbol{U}_1^T\ \ \boldsymbol{U}_2^T\begin{bmatrix}\boldsymbol{p}_{1k}\\ \boldsymbol{p}_{2k}\\ \boldsymbol{q}_{1k}\\ \boldsymbol{q}_{2k}\end{bmatrix} \tag{3.20}$$

In practice, the system described by (3.17) or (3.19) is very easy to implement. By putting (3.20) in $y(k)$, we have

$$\begin{aligned}y(k)&=\boldsymbol{w}_k^T\boldsymbol{x}_k\\ &=\left[\boldsymbol{p}_{1k}^T\ \ \boldsymbol{p}_{2k}^T\ \ \boldsymbol{q}_{1k}^T\ \ \boldsymbol{q}_{2k}^T\right]\boldsymbol{U}_2\boldsymbol{U}_1\boldsymbol{x}_k\\ &=\left[\boldsymbol{p}_{1k}^T\ \ \boldsymbol{p}_{2k}^T\ \ \boldsymbol{q}_{1k}^T\ \ \boldsymbol{q}_{2k}^T\right]\begin{bmatrix}\boldsymbol{s}_{p1,k}\\ \boldsymbol{s}_{p2,k}\\ \boldsymbol{s}_{q1,k}\\ \boldsymbol{s}_{q2,k}\end{bmatrix}\end{aligned} \tag{3.21}$$

where

$$\boldsymbol{s}_{pi,k}=[s_{pi,0}(k),s_{pi,1}(k),\cdots,s_{pi,N/2-1}(k)]^T \qquad i=1,2 \tag{3.22}$$

$$\boldsymbol{s}_{qi,k}=[s_{qi,0}(k),s_{qi,1}(k),\cdots,s_{qi,N/2-1}(k)]^T \qquad i=1,2 \tag{3.23}$$

The block diagram of a two-stage split-path adaptive system is shown in Fig. 3.2.

To recompose the four new input sequences for the respective subfilters, we simply need to multiply the composite "splitting matrix", i.e. $\boldsymbol{U}_2\boldsymbol{U}_1$, with the original signal vector, that is

$$\begin{bmatrix} \boldsymbol{s}_{p1,k} \\ \boldsymbol{s}_{p2,k} \\ \boldsymbol{s}_{q1,k} \\ \boldsymbol{s}_{q2,k} \end{bmatrix} = \boldsymbol{U}_2\boldsymbol{U}_1\boldsymbol{x}_k \tag{3.24}$$

Fig. 3.3 illustrates the procedure to compute (3.24) for an M=8 system graphically. Apparently, the realization of (3.24) is trivial, only $2M$ extra accumulators are needed for the hardware.

Now, the overall system output will be the sum of those of the subunits,

$$\begin{aligned} y(k) &= y_{p1}(k) + y_{p2}(k) + y_{q1}(k) + y_{q2}(k) \\ &= \boldsymbol{p}_{1k}^T \boldsymbol{s}_{p1,k} + \boldsymbol{p}_{2k}^T \boldsymbol{s}_{p2,k} + \boldsymbol{q}_{1k}^T \boldsymbol{s}_{q1,k} + \boldsymbol{q}_{2k}^T \boldsymbol{s}_{q2,k} \end{aligned} \tag{3.25}$$

and the ensemble output error is given by

$$\begin{aligned} e(k) &= d(k) - y(k) \\ &= d(k) - [\boldsymbol{p}_{1k}^T \boldsymbol{s}_{p1,k} + \boldsymbol{p}_{2k}^T \boldsymbol{s}_{p2,k} + \boldsymbol{q}_{1k}^T \boldsymbol{s}_{q1,k} + \boldsymbol{q}_{2k}^T \boldsymbol{s}_{q2,k}] \end{aligned} \tag{3.26}$$

The parameter updating equations for the branches are,

$$\boldsymbol{p}_{i,k+1} = \boldsymbol{p}_{ik} - \mu_{pi}(\nabla_{pi}\varepsilon)_k \qquad i = 1,2 \tag{3.27}$$

$$\boldsymbol{q}_{i,k+1} = \boldsymbol{q}_{ik} - \mu_{qi}(\nabla_{qi}\varepsilon)_k \qquad i = 1,2 \tag{3.28}$$

where ∇_{pi} and ∇_{qi} are the gradient operation with respect to $\boldsymbol{p}_{ik}$ and $\boldsymbol{q}_{ik}$ (i=1,2).

If the LMS algorithm is applied, and assume that all parameter vectors are independent to each other, we have

$$\boldsymbol{p}_{i,k+1} = \boldsymbol{p}_{ik} + 2\mu_{pi} e(k) \boldsymbol{s}_{pi,k} \qquad i = 1,2 \tag{3.29}$$

and

$$\boldsymbol{q}_{i,k+1} = \boldsymbol{q}_{ik} + 2\mu_{qi} e(k) \boldsymbol{s}_{qi,k} \qquad i = 1,2 \tag{3.30}$$

The subfilters are now adjusted simultaneously but independently to achieve their optimal solutions.

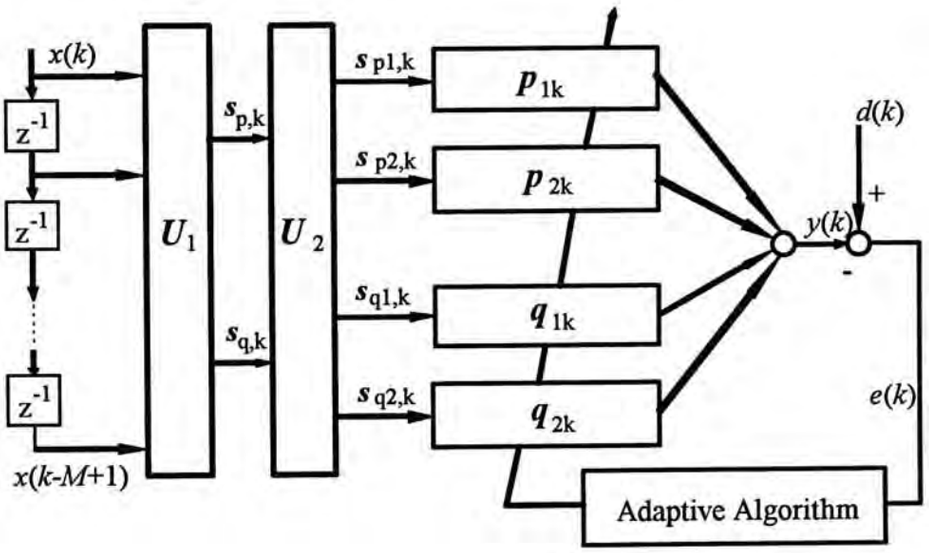


Fig. 3.2. A two-stage split-path adaptive filter.

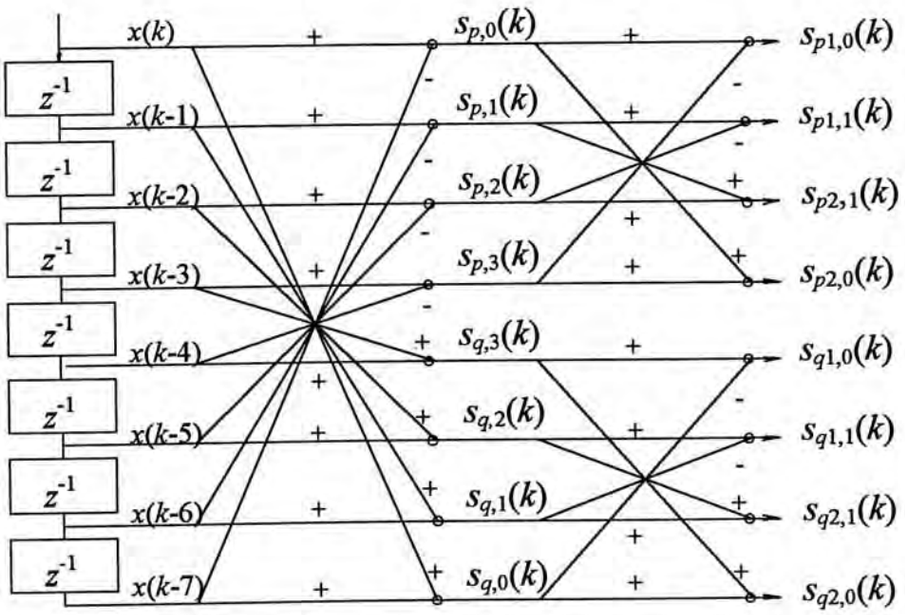


Fig. 3.3 Construction of the new input vector for a 2-stage split system (M=8).

3.3 Multi-Stage Split Structure for an FIR Adaptive Filter

In this section, a more general form of the multi-stage split structure adaptive system will be developed. We shall demonstrate that any $M=2^L$ coefficients FIR filter can be decomposed into M parallel single-parameter filters by employing the previously described symmetric/antisymmetric splitting operations. The new system parameters are being updated simultaneously by using a transformed input vector derived from the L-step splitting matrices [Wan 1995a]. Because the eigenvalue spread has been reduced during the splitting process, faster convergence speed can be achieved.

Let us define an $M\times M$ splitting block diagonal matrix as follows,

$$\boldsymbol{U}_i \triangleq \begin{bmatrix} \boldsymbol{u}_i & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \boldsymbol{u}_i \end{bmatrix} \qquad i = 1,\ 2,\ \cdots,\ L \tag{3.31}$$

This matrix has its diagonal element, $\boldsymbol{u}_i$, of the form

$$\boldsymbol{u}_i \triangleq \begin{bmatrix} \boldsymbol{I}_i & -\boldsymbol{J}_i \\ \boldsymbol{I}_i & \boldsymbol{J}_i \end{bmatrix} \qquad i = 1,\ 2,\ \cdots,\ L \tag{3.32}$$

where $\boldsymbol{I}_i$ and $\boldsymbol{J}_i$ are $(M/2^i)\times(M/2^i)$ identity matrix and anti-diagonal identity matrix. By applying the splitting matrices $\{\boldsymbol{U}_i,\ i=1,2...L\}$ to decompose the original system step by step, we ultimately get M branches that are connected in parallel. Each of these branches has only one single parameter and is therefore a simple adaptive gain unit. Let $c_i(k)$ denotes the parameter of the i^{th} filter branch, we can construct a new decomposed system which can be described as,

$$W(z) = \left[z^0, z^{-1},\ \cdots,\ z^{-M+1}\right] \boldsymbol{U}_1^T \boldsymbol{U}_2^T \cdots \boldsymbol{U}_L^T \begin{bmatrix} c_1(k) \\ c_2(k) \\ \vdots \\ c_M(k) \end{bmatrix} \tag{3.33}$$

where

$$\boldsymbol{c}_k = [c_1(k), c_2(k), \cdots, c_M(k)]^T \tag{3.34}$$

Hence, the original M-length adaptive filter is transformed into a collection of M-scalar independent adaptive subunits. In other words, the tapped delay line structure is transformed into an adaptive linear combiner. The parallel inputs for the linear combiner is then given by

$$\boldsymbol{g}_k = [g_1(k), g_2(k), \cdots, g_M(k)]^T \tag{3.35}$$

which can easily be generated from

$$\boldsymbol{g}_k = \boldsymbol{U}_L \boldsymbol{U}_{L-1} \cdots \boldsymbol{U}_1 \boldsymbol{x}_k \tag{3.36}$$

The combiner weighs and sums the branch inputs to form the system output,

$$y(k) = \boldsymbol{c}_k^T \boldsymbol{g}_k \tag{3.37}$$

The system is called a "full split adaptive system". This novel multi-stage or full split-path adaptive system is diagrammtically shown in Fig. 3.4.

Different adaptive algorithms can be used for the multi-stage split-path system. When the LMS algorithm is used, the equation to update the gain parameter vector of the system is given by

$$\boldsymbol{c}_{k+1} = \boldsymbol{c}_k + 2\mu\, e(k)\boldsymbol{g}_k \tag{3.38}$$

where the output error $e(k)$ can be computed from

$$e(k) = d(k) - \boldsymbol{c}_k^T \boldsymbol{g}_k \tag{3.39}$$

and μ is a convergence factor that controls stability and convergence of the system.

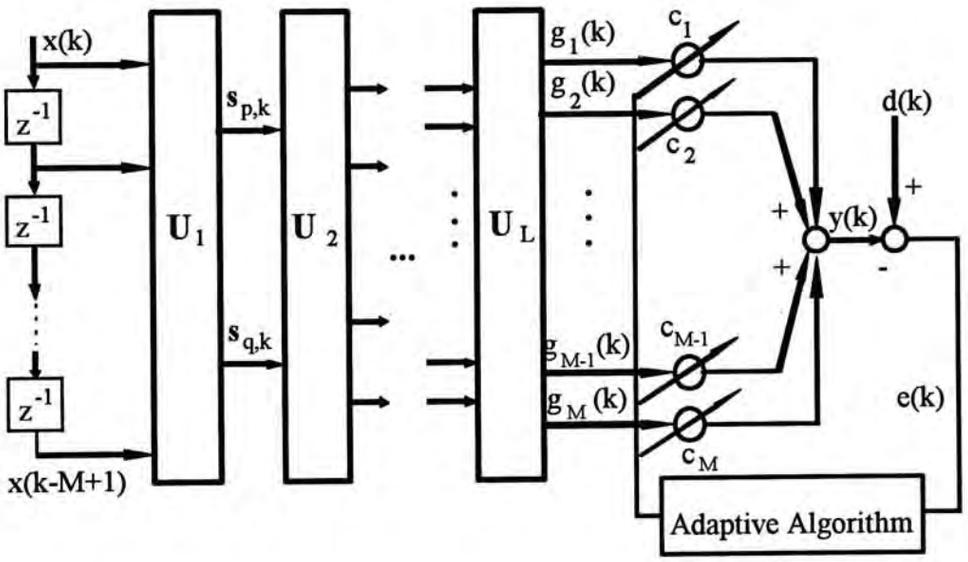


Fig. 3.4 Multi-stage split-path structure adaptive filter.

Because the branches are effectively uncorrelated to each other, different convergence factors can be employed for individual branches. That means the μ in (3.38) can be replaced by a matrix

$$\mu \triangleq \begin{bmatrix} \mu_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \mu_M \end{bmatrix} \tag{3.40}$$

where μ_i is the step size corresponding to the i^{th} branch. This is a major advantage of the split-path adaptive system. With more flexible choice of convergence factors for individual branches, further enhanced convergence behaviour can be anticipated.

In applications, if the parameters of the original structure is interested, for some particular cases such as system identification, it is easy to compute from the following parameter relation formula

$$\boldsymbol{w}_k = \boldsymbol{U}_1^T \boldsymbol{U}_2^T \cdots \boldsymbol{U}_L^T \boldsymbol{c}_k \tag{3.41}$$

3.4 Properties of the Split Structure LMS Algorithm

The mean square error (MSE) is commonly used as a cost function to measure the performance of an adaptive system. For a one step split operation, $e(k)$ is given by

$$e(k)=d(k)-\begin{bmatrix} \boldsymbol{p}_k^T & \boldsymbol{q}_k^T \end{bmatrix} \begin{bmatrix} \boldsymbol{s}_{p,k}^T \\ \boldsymbol{s}_{q,k}^T \end{bmatrix} \tag{3.42}$$

Squaring (3.42) and taking expectation yields

$$\varepsilon=\xi-2\boldsymbol{p}_k^T \boldsymbol{R}_{dp}-2\boldsymbol{q}_k^T \boldsymbol{R}_{dq}+(\boldsymbol{p}_k^T \boldsymbol{R}_p \boldsymbol{p}_k+\boldsymbol{q}_k^T \boldsymbol{R}_q \boldsymbol{q}_k)+2\boldsymbol{p}_k^T \boldsymbol{R}_{pq} \boldsymbol{q}_k \tag{3.43}$$

where $\xi = \mathbf{E}[d^2(k)]$, $\boldsymbol{R}_p = \mathbf{E}[\boldsymbol{s}_{p,k}\, \boldsymbol{s}^T_{p,k}]$ and $\boldsymbol{R}_q = \mathbf{E}[\boldsymbol{s}_{q,k}\, \boldsymbol{s}^T_{q,k}]$ are the autocorrelation matrices of $\boldsymbol{s}_{p,k}$ and $\boldsymbol{s}_{q,k}$, and $\boldsymbol{R}_{dp} = \mathbf{E}[d(k)\boldsymbol{s}_{p,k}]$, $\boldsymbol{R}_{dq} = \mathbf{E}[d(k)\boldsymbol{s}_{q,k}]$ and $\boldsymbol{R}_{pq} = \mathbf{E}[\boldsymbol{s}_{p,k}\, \boldsymbol{s}^T_{q,k}]$ are the cross-correlation matrices. From (2.34), we have

$$\boldsymbol{E}\left(\begin{bmatrix}\boldsymbol{s}_{p,k}\\ \boldsymbol{s}_{q,k}\end{bmatrix}\begin{bmatrix}\boldsymbol{s}^T_{p,k} & \boldsymbol{s}^T_{q,k}\end{bmatrix}\right)=\begin{bmatrix}\boldsymbol{R}_p & \boldsymbol{R}_{pq}\\ \boldsymbol{R}_{pq} & \boldsymbol{R}_q\end{bmatrix}=\boldsymbol{U}_1\boldsymbol{R}_x\boldsymbol{U}_1^T \tag{3.44}$$

Note that the input autocorrelation matrix $\boldsymbol{R}_x$ is a symmetric Toeplitz matrix where all the elements along each diagonal are equal [Gray, 1972] which can be expressed in a block matrix form

$$\boldsymbol{R}_x=\begin{bmatrix}\boldsymbol{R}_0 & \boldsymbol{R}_1^T\\ \boldsymbol{R}_1 & \boldsymbol{R}_0\end{bmatrix} \tag{3.45}$$

Since $\boldsymbol{R}_0$ is also a symmetric Toeplitz matrix, it is easy to show that

$$\boldsymbol{J}_1\boldsymbol{R}_1=\boldsymbol{R}_1^T\boldsymbol{J}_1 \qquad \text{and} \qquad \boldsymbol{J}_1\boldsymbol{R}_0\boldsymbol{J}_1=\boldsymbol{R}_0 \tag{3.46}$$

Substituting (3.45) and (3.46) into (3.44), we have

$$\begin{aligned}\begin{bmatrix}\boldsymbol{R}_p & \boldsymbol{R}_{pq}\\ \boldsymbol{R}_{pq} & \boldsymbol{R}_q\end{bmatrix}&=\begin{bmatrix}\boldsymbol{I}_1 & -\boldsymbol{J}_1\\ \boldsymbol{I}_1 & \boldsymbol{J}_1\end{bmatrix}\begin{bmatrix}\boldsymbol{R}_0 & \boldsymbol{R}_1^T\\ \boldsymbol{R}_1 & \boldsymbol{R}_0\end{bmatrix}\begin{bmatrix}\boldsymbol{I}_1 & \boldsymbol{I}_1\\ -\boldsymbol{J}_1 & \boldsymbol{J}_1\end{bmatrix}\\ &=\begin{bmatrix}2\boldsymbol{R}_0-2\boldsymbol{J}_1\boldsymbol{R}_1 & 0\\ 0 & 2\boldsymbol{R}_0+2\boldsymbol{J}_1\boldsymbol{R}_1\end{bmatrix}\end{aligned} \tag{3.47}$$

That is $\boldsymbol{R}_{pq}=0$. This proves that $\boldsymbol{s}_{p,k}$ and $\boldsymbol{s}_{q,k}$ are totally independent to each other. From (3.47), we have found that the splitting operation not only reconstructs the input vectors, but also compresses the autocorrelation matrix $\boldsymbol{R}_x$ as well. More precisely, it

has a unique function to diagnolize $\boldsymbol{R}_x$. As a result, the split subsystems are completely decoupled. Therefore, (3.43) can be simplified to

$$\varepsilon = \xi - 2\boldsymbol{p}_k^T \boldsymbol{R}_{dp} - 2\boldsymbol{q}_k^T \boldsymbol{R}_{dq} + (\boldsymbol{p}_k^T \boldsymbol{R}_p \boldsymbol{p}_k + \boldsymbol{q}_k^T \boldsymbol{R}_q \boldsymbol{q}_k) \tag{3.48}$$

In this case, adaptation of each parameter vector can be done simultaneously and independently. The updating equations are as follows,

$$\boldsymbol{p}_{k+1} = \boldsymbol{p}_k + 2\mu_p \, e(k)\boldsymbol{s}_{p,k} \tag{3.49}$$

and

$$\boldsymbol{q}_{k+1} = \boldsymbol{q}_k + 2\mu_q \, e(k)\boldsymbol{s}_{q,k} \tag{3.50}$$

where the control factors, μ_p and μ_q, can be assigned with different values for different subfilters to further speed up the convergence rate.

For the system with two splitting stages, we have

$$E\left(\begin{bmatrix} \boldsymbol{s}_{p1,k} \\ \boldsymbol{s}_{p1,k} \\ \boldsymbol{s}_{q1,k} \\ \boldsymbol{s}_{q2,k} \end{bmatrix} \begin{bmatrix} \boldsymbol{s}_{p1,k}^T & \boldsymbol{s}_{p2,k}^T & \boldsymbol{s}_{q1,k}^T & \boldsymbol{s}_{q2,k}^T \end{bmatrix}\right) = \boldsymbol{U}_2 \boldsymbol{U}_1 \boldsymbol{R}_{\mathrm{x}} \, \boldsymbol{U}_1^T \boldsymbol{U}_2^T \tag{3.51}$$

By using the result in (3.47) and invoking the definition of $\boldsymbol{U}_2$ yields

$$\begin{aligned} \boldsymbol{U}_2 \boldsymbol{U}_1 \boldsymbol{R}_{\mathrm{x}} \, \boldsymbol{U}_1^T \boldsymbol{U}_2^T &= \boldsymbol{U}_2 \begin{bmatrix} \boldsymbol{R}_p & 0 \\ 0 & \boldsymbol{R}_q \end{bmatrix} \boldsymbol{U}_2^T \\ &= \begin{bmatrix} \boldsymbol{u}_2 \boldsymbol{R}_p \boldsymbol{u}_2^T & 0 \\ 0 & \boldsymbol{u}_2 \boldsymbol{R}_q \boldsymbol{u}_2^T \end{bmatrix} \end{aligned} \tag{3.52}$$

In (3.52), $\boldsymbol{u}_2$ provides a data compression operation on $\boldsymbol{R}_p$ and $\boldsymbol{R}_q$, respectively, which is similar to what $\boldsymbol{U}_1$ has done on $\boldsymbol{R}_x$.

While for a full split system, it is also easy to show that the autocorrelation matrix $\boldsymbol{R}_g=\mathbf{E}[\boldsymbol{g}_k\boldsymbol{g}^T_k]$ strictly retains the form as in (3.47). That is, $\boldsymbol{R}_g$ can be divided into four subblocks and its top right and bottom left subblocks are always null matrices. By the compression operation made by $\boldsymbol{u}_i$, $\boldsymbol{R}_g$ will gradually approach to a diagonal form.

The above result can be considered from another point of view. Since $\boldsymbol{U}_i$ is an orthogonal matrix, the product $\boldsymbol{U}_L\boldsymbol{U}_{L-1}...\boldsymbol{U}_1$ is actually an orthogonal transform applied onto $\boldsymbol{x}_k$. It has been proved that when the decaying time constant of the input autocorrelation $\boldsymbol{R}_x$ is sufficiently small compared to the filter length, $\boldsymbol{R}_x$ can be approximated by a circulant matrix such that $\boldsymbol{R}_g$ is close to a diagonal form [Lee, 1986]. In (3.47), $\boldsymbol{U}_1\boldsymbol{R}_x\boldsymbol{U}^T_1$ is exactly a diagonal block matrix. When further splitting operations $\boldsymbol{U}_i$ (i=2,... L) are applied, the autocorrelation matrix can still be expressed in a diagonal block format. With the increasing of i, the non-zero elements in $\boldsymbol{R}_g$ will only distribute to the nearby elements of the diagonal. Thus the principal elements will gradually concentrate to the diagonal of the autocorrelation matrix. This means that the correlation between M parallel inputs is getting weaker and $\boldsymbol{R}_g$ becomes more diagonally dominant. An original M-length adaptive system is reduced to a set of M decoupled adaptive subunits each with a scalar parameter. Computer simulations have also confirmed that the "principal values" will always locate on the diagonal of $\boldsymbol{R}_g$ after splitting.

Define a parameter estimate error vector for the full split system

$$\boldsymbol{v}_k=\boldsymbol{c}_k-\boldsymbol{c}^* \tag{3.53}$$

where $\boldsymbol{c}^*$ is assumed to be the optimal weight vector for the full split system. Put (3.53) in (3.38), we have

$$\begin{aligned} \boldsymbol{v}_{k+1} &= \boldsymbol{v}_k + 2\mu\ e(k)\boldsymbol{g}_k \\ &= \boldsymbol{v}_k + 2\mu\ (\boldsymbol{g}_k^T\boldsymbol{c}^* - \boldsymbol{g}_k^T\boldsymbol{c})\boldsymbol{g}_k \end{aligned} \tag{3.54}$$

Taking expectation of (3.54) yields

$$\begin{aligned} E[\boldsymbol{v}_{k+1}] &= E[\boldsymbol{v}_k] + 2\mu E[\ (\boldsymbol{g}_k^T\boldsymbol{c}^* - \boldsymbol{g}_k^T\boldsymbol{c})\boldsymbol{g}_k\ ] \\ &= E[\boldsymbol{v}_k] - 2\mu E[(\ \boldsymbol{g}_k\boldsymbol{g}_k^T)\boldsymbol{v}_k\ ] \end{aligned} \tag{3.55}$$

Assume $\boldsymbol{v}_k$ changes very slowly during adaptation (this assumption is commonly used in most literatures), we have,

$$\begin{aligned} E[\boldsymbol{v}_{k+1}] &= E[\boldsymbol{v}_k] - 2\mu E[(\ \boldsymbol{g}_k\boldsymbol{g}_k^T\ )\boldsymbol{v}_k\ ] \\ &= E[\boldsymbol{v}_k] - 2\mu \boldsymbol{R}_g E[\boldsymbol{v}_k\ ] \\ &= (\boldsymbol{I} - 2\mu \boldsymbol{R}_g)E[\boldsymbol{v}_k\ ] \end{aligned} \tag{3.56}$$

When self-orthogonal adjustment LMS algorithm [Gitlin, 1977] is introduced, (3.56) becomes

$$E[\boldsymbol{v}_{k+1}] = (\boldsymbol{I} - 2\mu\ \Lambda^{-2}\boldsymbol{R}_g)E[\boldsymbol{v}_k\ ] \tag{3.57}$$

and a full split-path self-orthogonal LMS algorithm is obtained,

$$\boldsymbol{c}_{k+1} = \boldsymbol{c}_k + 2\mu\ \Lambda^{-2}e(k)\boldsymbol{g}_k \tag{3.58}$$

where $\Lambda^2 = \mathrm{diag}\{\sigma_1^2, \sigma_2^2, \ldots \sigma_M^2\}$, and σ_i^2 is the power estimate for branch i that can be computed by averaging with a moving window. Since $\boldsymbol{R}_g$ is approximately diagonal, it is actually an approximation of the power spectrum and Λ^2 can be taken as the estimate of $\boldsymbol{R}_g$. Therefore, $\Lambda^{-2}\boldsymbol{R}_g$, in equation (3.57), will approach to unity, the eigenvalue spread of the associated transformed signal matrix is approximately unity as well, therefore, a faster convergence speed can be expected.

Another advantage of the split structure is its relative insensitivity to parameter fluctuations. For an ordinary high order transversal filter, the adaptation characteristics are in general sensitive to variation of coefficients. However, for a split structure adaptive filter, the high order system is being reduced into a set of low order subsystems, which in turn reduces the sensitivity to parameter fluctuation, therefore increases the stable operation range of the overall adaptive system.

The generalization of the continuous split algorithm is easy to implement. For a typical system with 8 parameters, the full split input vector is formed by $\boldsymbol{U}_3\boldsymbol{U}_2\boldsymbol{U}_1$

$$\boldsymbol{g}_k = \boldsymbol{U}_3\boldsymbol{U}_2\boldsymbol{U}_1\boldsymbol{x}_k = \begin{bmatrix} 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \boldsymbol{x}_k = \boldsymbol{H}_8\boldsymbol{x}_k \quad (3.59)$$

The flow diagram for calculating (3.59) is shown in Fig. 3.5. Comparing $\boldsymbol{H}_8$ with an 8×8 Walsh-Hadamard transform matrix [Wan 1992], we can see that they have exactly the same elements except for some permutation changes in rows. This gives us some very important revelation of the characteristics of the split structure. The unification between split-path structure with discrete Walsh transform (DWT) will be discussed in depth in Chapter 6.

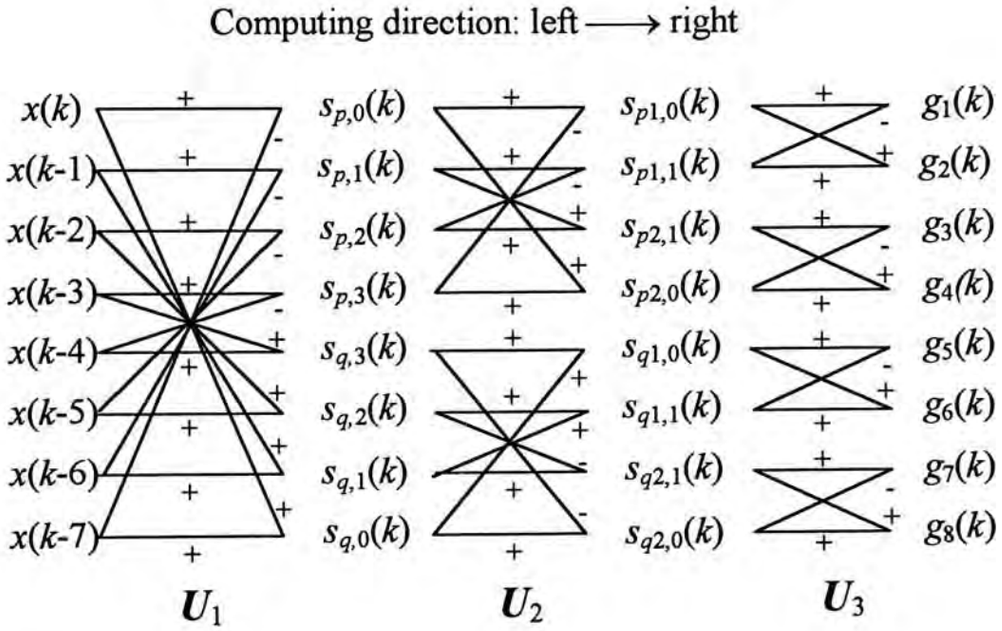


Fig. 3.5 Flow chart for computing parallel input.

3.5 Full Split-Path Adaptive Algorithm for System Identification

An application of the proposed algorithm to adaptive system identification is shown in Fig. 3.6 which is used to evaluate the performance of the split algorithm and to validate its unification with DWT. As an example, a 7^{th}-order plant was to be identified which had the following transfer function,

$$\begin{aligned} H(z) = {} & 0.97 + 1.224z^{-1} + 0.429z^{-2} + 1.141z^{-3} \\ & + 0.674z^{-4} + 0.265z^{-5} + 2.208z^{-6} + 3.071z^{-7} \end{aligned} \tag{3.60}$$

The input sequence, $x(k)$, had a power fixed at $\sigma_x^2=1$. The additive noise $n(k)$ was treated as the measuring error while the sum of the plant output and $n(k)$ becomes the desired output signal. By changing the system model within the block marked by dotted line as shown in Fig. 3.6, different adaptive algorithms could be applied. All simulation results were obtained from an average of 500 independent runs. Fig. 3.7-3.9 compare the learning curves of four different types of adaptive structures, namely the non-split configuration, the 1-level, 2-level and full split systems. It is shown that the performance of split algorithms is superior than the conventional LMS method in almost all conditions. By splitting up the system continuously into a parallel of sub-sections, the convergence behaviour is improved and the full split structure always achieves the fastest convergence speed. It is also noted that a full split-path adaptive filter has identical performance as an 8×8 DWT adaptation. This demonstrates the unification of split-path adaptive filtering and adaptation in Walsh transform domain.

It is well known that the convergence factor μ takes an important role in controlling the convergence speed and stability, a bigger μ can give a faster convergence speed but it will also produce a greater level of fluctuation to the output of the adaptive system. Fig. 3.10 compares the learning curves for different adaptive structures when the convergence factor was as large as 0.075. It is observed that under this situation, there was a serious fluctuation in the MSE of the output for the conventional LMS method. But such fluctuation decreases if the split-path technique is incorporated and the full split structure will have the smallest fluctuation in MSE. The reason being that the split operation tries to whiten the input signal before adaptation, thereby providing a wider admissible range for the convergence factor. This also demonstrates that the split-path adaptive system is less sensitive to fluctuation of parameters as the input signal has greater level of changes. Hence, the split-path system possesses a more robust stability behaviour.

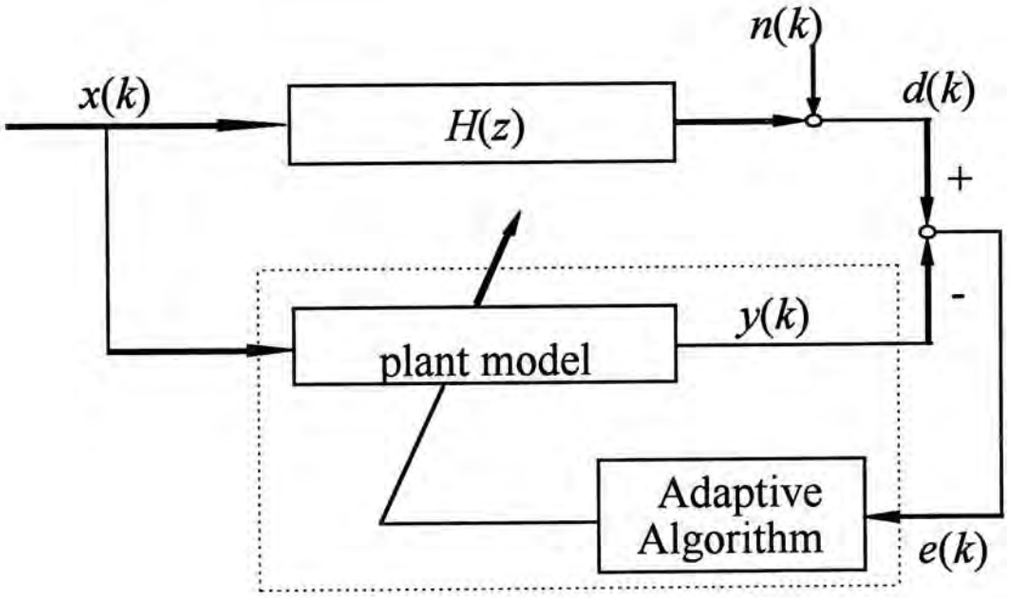


Fig. 3.6 Adaptive system identification model.

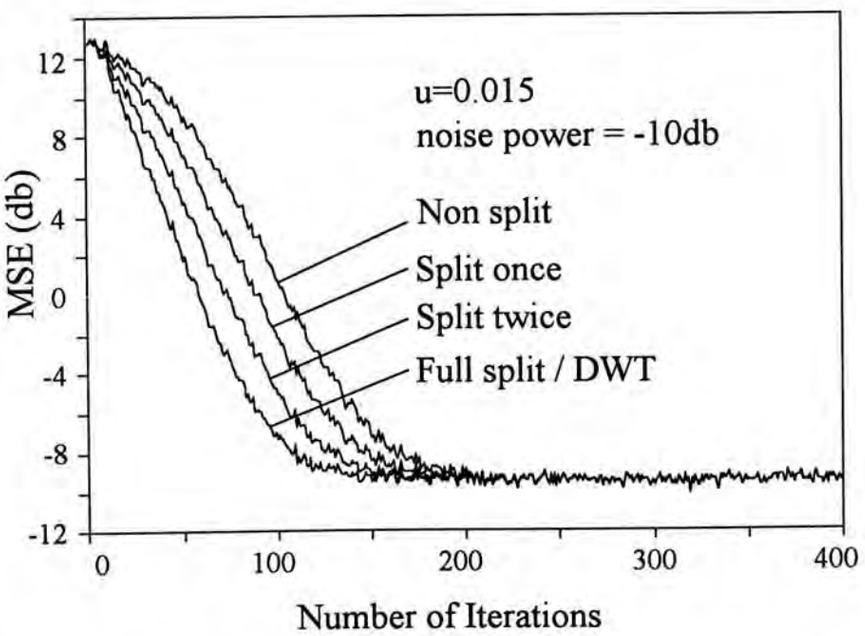


Fig. 3.7 Comparison of learning curves (noise power = -10db).

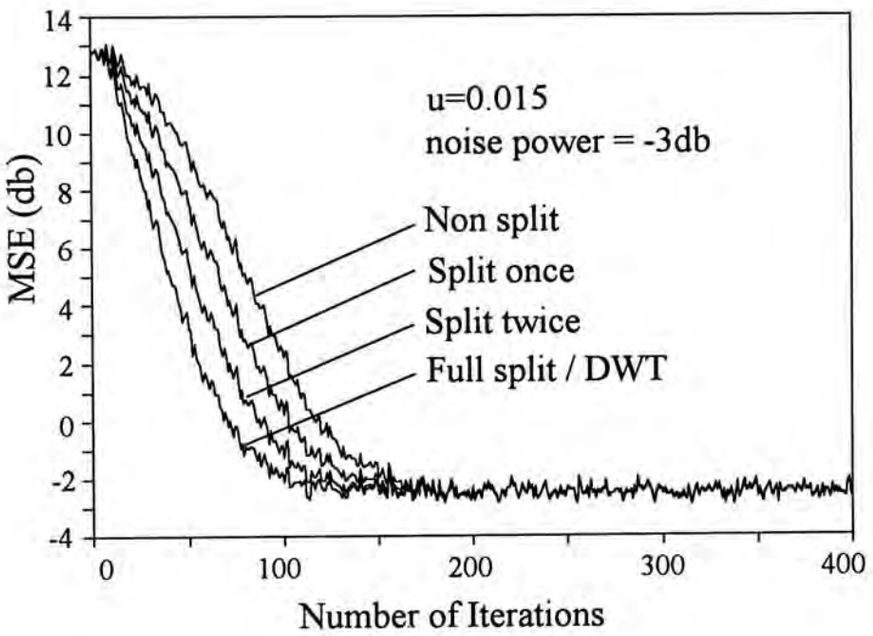


Fig. 3.8 Comparison of learning curves (noise power = -3db).

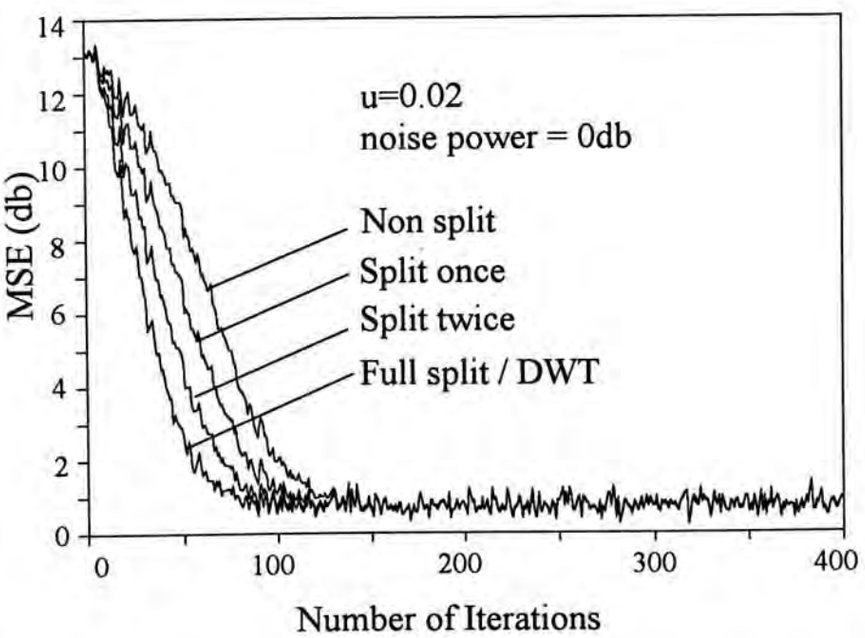


Fig. 3.9 Comparison of learning curves (noise power = 0db).

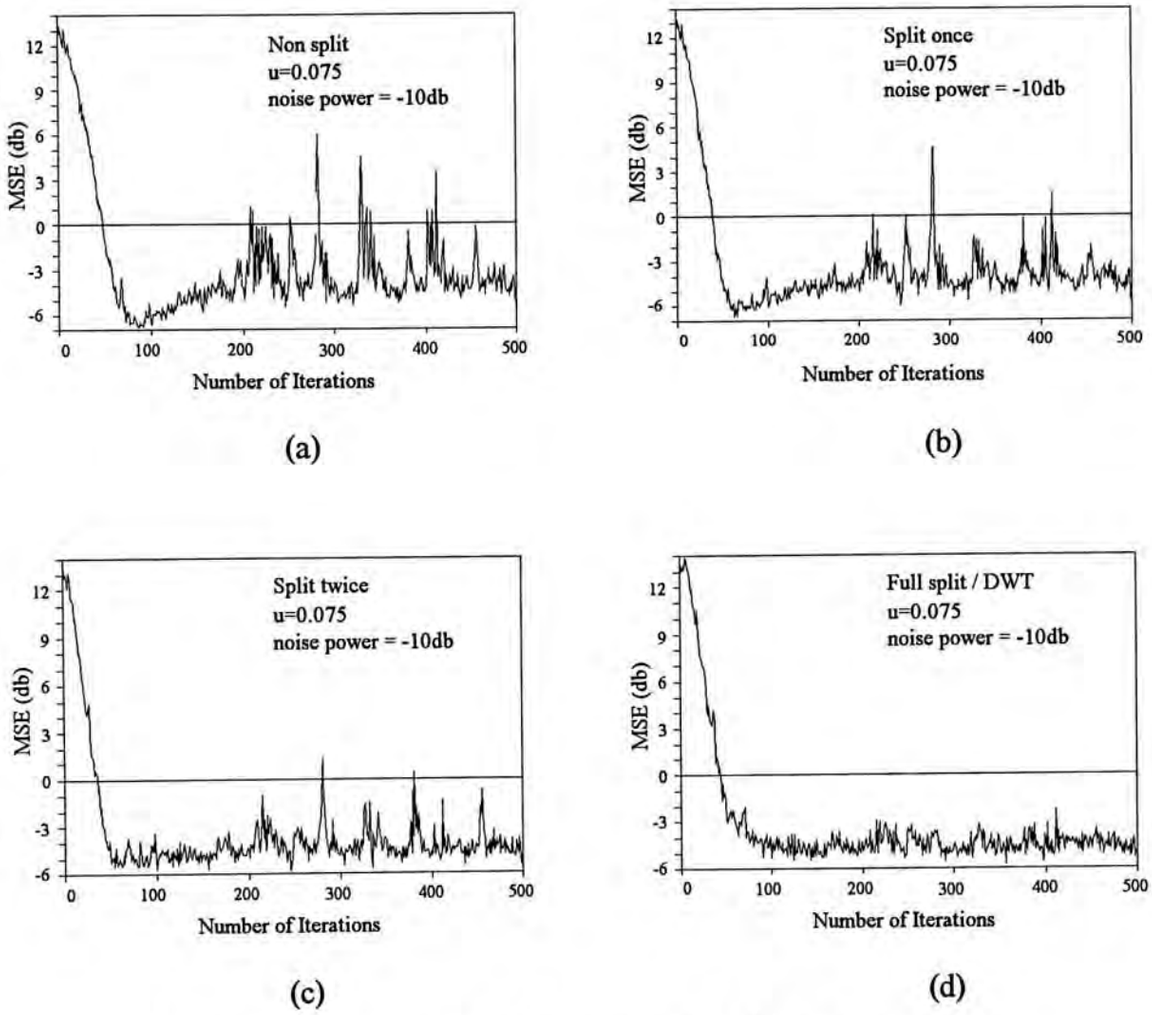


Fig. 3.10 Comparison of fluctuations in MSE with μ=0.075:

(a). non-split structure; (b). one-split structure;

(c). two-split structure; (d). full-split structure.

3.6 Summary

In this chapter, we first demonstrate that how a FIR can be decomposed in a pragmatic way by using the split technique. We then develop a generalized modular form of multi-stage split-path structure for the adaptive systems. We show that any $M=2^L$-coefficient system can be decomposed, step by step, into a collection of M parallel single parameter subfilters by L split operations. The properties of the split-path adaptive LMS algorithm are investigated. The essential function of the split method is its capability for diagonalizing the input autocorrelation matrix, thereby reducing the eigenvalue spreads associated with subsystems. This makes the split algorithm an efficient and effective method to improve the performance of adaptive filtering.

Besides, the multi-stage split-path structure adaptive filtering is easy to realize, only $L \times M$ extra additions are needed for the split operation to be implemented.

Chapter 4 TRANSFORM DOMAIN SPLIT-PATH ADAPTIVE ALGORITHMS

In the past few decades, there has been an increasing interest with respect to using a class of orthogonal transforms in data compression and other digital signal processing applications. Various unitary transforms have attracted considerable attention for their applications in transform domain adaptive algorithms and have been shown effective to increase the convergence speed by whitening the power spectrum of the input signal. Because the autocorrelation matrix of the transformed signal is *approximately* whitened, the potential for further enhancement of the performance of the adaptive system still exits. In this chapter, a transform domain split-path adaptive algorithm is investigated. The properties and performance characteristics of the transform domain split-path adaptive filter (TRSPAF) are examined thoroughly. The optimum Wiener solution and the steady state MSE of the adaptive system are derived. The proposed structure allows different convergence factors to be used for the two filter paths in split structure to achieve a superior convergence behaviour. The analysis is general and rigorous, and does not restrict to any particular transformation. Simulation results show that the performance of the split-path predictor can be improved remarkably when adaptation is performed in the transform domain while its system complexity still remains unchanged [Wan 1995b].

4.1 Introduction

Adaptive filters are generally implemented in the time domain and particularly the LMS algorithm is widely used in many applications because of its simplicity in realization. One drawback of the LMS algorithm is that its convergence speed decreases as the eigenvalue spread of the input autocorrelation matrix increases. An ideal approach to increase the convergence speed of the LMS algorithm is to use a matrix convergence factor that is obtained by multiplying the inverse of the input autocorrelation matrix by a scalar constant. This type of weight adjustment is known as self-orthogonalization [Gitlin 1977] [Widrow 1984]. It can be shown that the resultant matrix controlling the convergence speed of this algorithm is an identity matrix and, thus the eigenvalues are all equal. However, realization of this algorithm required a large number of arithmetic operations due to matrix manipulations.

Some reports have pointed out that adaptive filters can be implemented in frequency domain to achieve faster convergence speed [Dentino 1978] [Bershad 1979] [Reed 1981] [Bitmead 1981]. The fast Fourier transform (FFT) algorithm can also be used to reduced the computational requirement for implementing the TDL adaptive filters in transform domain [Ferrara 1980].

Based on the fact that eigenvalue spread is bounded by the ratio of the maximum to minimum magnitudes of the input power spectrum [Gersho 1969], Narayan *et al.* [1981] [1983] proposed a transform domain LMS to improve the convergence speed of the adaptive systems. The fundamental concept of the technique involves transforming the input sequence to some less correlated samples before updating the filter parameters. Theoretically speaking, any orthogonal transformation

can be used, of course, implementation complexity of different transforms may not be the same for every case. Furthermore, the ability of whitening incoming signals may also vary for different transforms and is usually dependent on signal characteristics. It has been illustrated in many literatures [Narayan 1983] [Lee 1986] [Ho 1992b] [Wan 1992] that transform domain adaptive filtering (TRAF) can improve the convergence rate substantially but in the expense of an increase in computation.

Since the whitened signal in transform domain is *nearly* uncorrelated, the relative disparity in eigenvalues still exists. In this chapter, we shall exploit the transform domain adaptive technique to a split-path structure linear prediction so that its convergence and tracking capabilities can further be improved. The properties and performance of the transform domain split-path adaptive filter are examined thoroughly. The optimum Wiener solution and the steady state MSE of the adaptive system are derived. The analysis is general and does not restrict to any particular transformation. Simulation results show that the performance of the split-path predictor can be improved remarkably when adaptation is performed in the transform domain while its system complexity still remains unchanged [Wan 1994a] [Wan 1995b].

4.2 General Description of Transforms

It is known that all eigenvalues of the input autocorrelation matrix $\boldsymbol{R}_x$ are bounded by the minimum and maximum values of the power spectrum of the input [Gersho 1969]. Therefore, one method to increase the convergence speed is to whiten

the power spectrum of the input signal by transformation. Let $\boldsymbol{x}_k$ be an $M\times1$ signal vector at time instant k,

$$\boldsymbol{x}_k = [x(k),\ x(k-1),\ \cdots,\ x(k-M+1)]^T \tag{4.1}$$

then its $M\times1$ transform vector $\boldsymbol{f}_k$, is given by,

$$\boldsymbol{f}_k = \boldsymbol{V}\boldsymbol{x}_k \tag{4.2}$$

where

$$\boldsymbol{f}_k = [f_0(k),\ f_1(k),\ \cdots,\ f_{M-1}(k)]^T \tag{4.3}$$

$\boldsymbol{V}$ denotes a unitary two-dimensional $M\times M$ transform matrix. The restriction of $\boldsymbol{V}$ to unitary matrices ensures conservation of signal energy in the transform domain, i.e.,

$$\sum_i |x(k-i)|^2 = \sum_i |f_i(k)|^2 \tag{4.4}$$

4.2.1 Fast Karhunen-Loeve Transform

Among all transforms, the Karhunen-Loeve transform (KLT) is known to be optimal in the sense that it yields uncorrelated data, simplifying succeeding operations. The KLT of $\boldsymbol{x}_k$ is a matrix $\boldsymbol{V}$, composed of the eigenvectors of $\boldsymbol{R}_x$ and is defined by the relation [Jain 1974],

$$\boldsymbol{V}\boldsymbol{R}_x\boldsymbol{V}^T = \Lambda^2 \tag{4.5}$$

where Λ^2 is a diagonal matrix of eigenvalues σ_i^2. When $x(k)$ is a first-order Markov process with zero mean and unit variance and the autocorrelation function is given by

$$E[x(k)x(k+n)] = \rho^{|n|} \qquad k = 0,1,\ \cdots,\ M+1 \tag{4.6}$$

then the KLT of the vector $\boldsymbol{x}_k$ can be computed from

$$\begin{aligned} f_i(k) &= \sum_{j=1}^{M} v_{ij} x(k-j) \\ &= \sum_{j=1}^{M} a_j x(k-j) \sin\left[\omega_j (i+\frac{M+1}{2}) + \frac{j\pi}{2}\right] \qquad i = 1, \cdots, M \end{aligned} \tag{4.7}$$

where

$$\sigma_i^2 = \frac{1-\rho^2}{1-2\rho\cos\omega_i + \rho^2} \tag{4.8}$$

and a_i is the normalization constant and $\{\omega_i\}$ are the positive roots of

$$\tan M\omega = -\frac{(1-\rho^2)\sin\omega}{\cos\omega - 2\rho + \rho^2\cos\omega} \tag{4.9}$$

Since the samples $f_i(k)$ are uncorrelated, they can be quantized independently. KLT also possesses minimum mean-square error property which makes it optimal for data compression [Watanabe 1965]. However, due to nonharmonicity of the sine terms in (4.7), fast algorithm is not available in computing the transform. In practice, implementation of the KLT is a formidable task when the block size or the number of samples is large. Generation of a transform matrix is involved, and in general, a large number of multiplications are required for each incoming signal vector. Therefore, some suboptimal transforms have usually been substituted for the KLT in the hope that they may somehow simulate the KLT.

For finite first-order stationary Gauss-Markov processes a fast Karhunen-Loeve transform (FKLT) was developed by Jain [Jain 1976] which is described as

$$f_i(k) = \sqrt{\frac{2}{M+1}} \sum_{j=1}^{M} x(k-j) \sin\frac{ij\pi}{M+1} \qquad i = 1, \cdots, M \tag{4.10}$$

This FKLT can be implemented via an FFT algorithm.

4.2.2 Symmetric Cosine Transform

The symmetric cosine transform (SCT) is derived similar to FKLT, which can be expressed as [Kitajima 1980],

$$f_i(k) = \sqrt{\frac{2}{M-1}}\, u(i) \sum_{j=0}^{M-1} u(j)x(k-j)\cos\frac{ij}{M-1}\pi \qquad i = 0,1,\cdots,M-1 \quad (4.11)$$

where

$$u(i) = \begin{cases} \frac{1}{\sqrt{2}} & i = 0, M-1 \\ 1 & \text{elsewhere} \end{cases} \quad (4.12)$$

The SCT has the property of asymptotic equivalence to the KLT. Because SCT is characterized by a symmetric matrix, both the forward and reverse operations can be handled by a single apparatus in hardware implementation [Kitajima 1980].

4.2.3 Discrete Sine Transform

Yip and Rao [Yip 1980] have proven that for large sequence length ($M \geq 32$) and low correlation coefficient, the discrete sine transform (DST) performs better to simulate KLT, on the basis of the residual correlation and of the rate distortion. Therefore, the DST is sometimes used as a substitute for KLT. A fast DST is described by [Wang 1982],

$$f_i(k) = c(i)\sqrt{\frac{2}{N}} \sum_{j=0}^{M-1} (-1)^j x(k-j)\sin\frac{(2j+1)i\pi}{2M} \qquad i = 0,1,\cdots,M\text{-}1 \quad (4.13)$$

where

$$c(i) = \begin{cases} \frac{\sqrt{2}}{2} & i = M-1 \\ 1 & \text{elsewhere} \end{cases} \tag{4.14}$$

4.2.4 Discrete Cosine Transform

The discrete cosine transform (DCT) is defined by [Ahmed 1974],

$$\begin{aligned} f_0(k) &= \frac{\sqrt{2}}{M} \sum_{j=0}^{M-1} x(k-j) \\ f_i(k) &= \frac{2}{M} \sum_{j=0}^{M-1} x(k-j) \cos\left[\frac{\pi(2j+1)i}{2M}\right] \qquad i = 0, \cdots, M-1 \end{aligned} \tag{4.15}$$

The DCT has gained widespread acceptance as a data compression method owing to its asymptotic equivalence to the KLT for first-order Markov random data. Algorithms for computing the DCT fall into two categories, one is based on other trigonometric transforms, such as the discrete Fourier transform (DFT) or discrete Hartley transform, and another computes the DCT directly [Heideman 1992].

4.2.5 Discrete Hartley Transform

The discrete Hartley transform (DHT), first published in 1942, is defined as [Bondyopadhyay 1988],

$$f_i(k) = \frac{1}{\sqrt{M}} \sum_{j=0}^{M-1} x(k-j) \operatorname{cas}(2\pi ij / M) \tag{4.16}$$

where

$$\operatorname{cas}(2\pi ij / M) = \cos(2\pi ij / M) + \sin(2\pi ij / M) \tag{4.17}$$

DHT is an alternative formulation of a harmonic functional transform similar to the Fourier identity. It can be obtained from the Fourier integral by replacing the exponential function $\exp(-j\omega t) = \cos(\omega t)-j\sin(\omega t)$ by $\text{cas}(\omega t) = \cos(\omega t)+ \sin(\omega t)$. As the DHT and DFT are closely related, the Fourier spectrum can be calculated via the Hartley transform. For real signals, the even and odd parts of the Hartley spectrum are the real and imaginary parts of the Fourier spectrum, respectively [Meckelburg 1985].

4.2.6 Discrete Walsh Transform

The discrete Walsh transform (DWT) is defined for a series of $M=2^L$ terms as [Beauchamp 1984],

$$f_i(k) = \frac{1}{M}\sum_{j=0}^{M-1} x(k-j)\text{Wal}(i,j) \tag{4.18}$$

where $\text{Wal}(i,j)$ is the (i, j)th element of DWT matrix, which is determined by

$$\text{Wal}(i,j) = \prod_{r=0}^{L-1}(-1)^{i_{L-1-r}(j_r+j_{r+1})} \tag{4.19}$$

where i and j are expressed in terms of their binary digits, e.g., $i=(i_{L-1}, i_{L-2},\ldots, i_1,i_0)$. Walsh function can be derived by different methods, each having its own property. DWT is a real number transform, its transform matrix components take on values of 1 and -1 only. The regular to compute DWT and inverse DWT are similar with only $1/M$ factor difference and its amount of computation is probably the fewest among all transforms. This distinguished feature makes the DWT most suitable for practical application.

4.3 Transform Domain Adaptive Filters

4.3.1 Structure of Transform Domain Adaptive Filter

The block diagram of a typical transform domain adaptive filter (TRAF) is shown in Fig. 4.1. The time domain input vector, $\boldsymbol{x}_k$, is first converted into a transform domain vector, $\boldsymbol{f}_k$, of the same length which can be expressed as

$$\boldsymbol{f}_k = \boldsymbol{V}\boldsymbol{x}_k \tag{4.20}$$

where $\boldsymbol{V}$ is an unitary matrix of rank M which represents the M-point discrete orthogonal transformation. As an updated $x(k)$ arrives, the transform domain input vector $\boldsymbol{f}_k$ is renewed as well. In general, if $\boldsymbol{V}$ is not a unitary matrix, then

$$\boldsymbol{V}\boldsymbol{V}^T = K\boldsymbol{I} \tag{4.21}$$

where K is a constant that might vary with the vector size M and the kind of transformation used. Table 4.1. lists several common used discrete transform, their transform coefficients and associated factor K.

By weighting the parameters in transform domain, where

$$\boldsymbol{\omega}_k = \left[\omega_0(k),\ \ \omega_1(k),\ \ \cdots,\ \ \omega_{M-1}(k)\right]^T \tag{4.22}$$

we can get the adaptive output.

$$y(k) = \sum_{i=0}^{M-1} \omega_i(k) f_i(k) = \boldsymbol{\omega}_k^T \boldsymbol{f}_k \tag{4.23}$$

Fig. 4.2 illustrates the realization of a transform domain adaptive filtering.

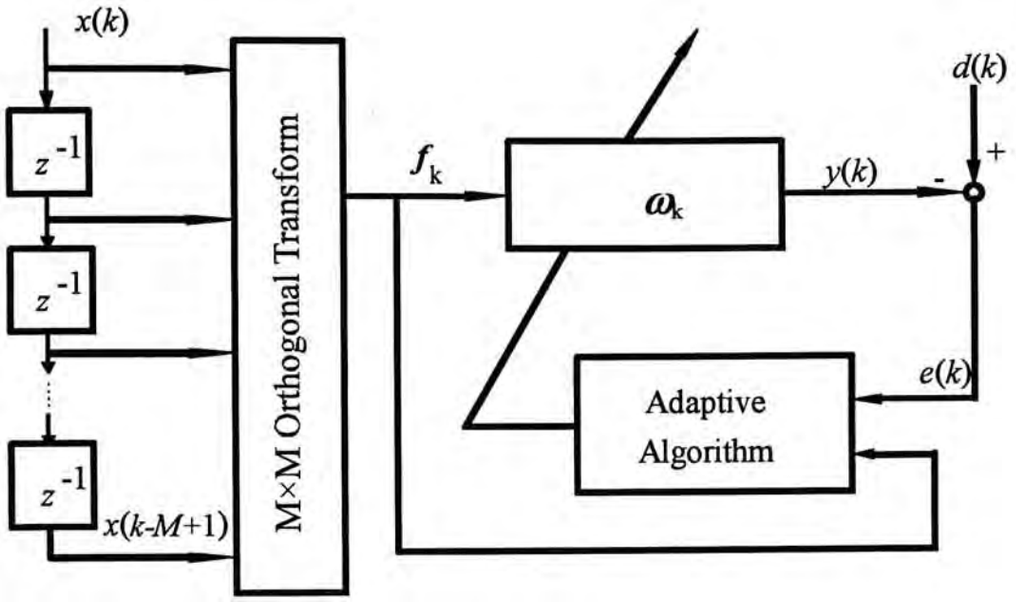


Fig. 4.1 Block diagram of a transform domain adaptive filter.

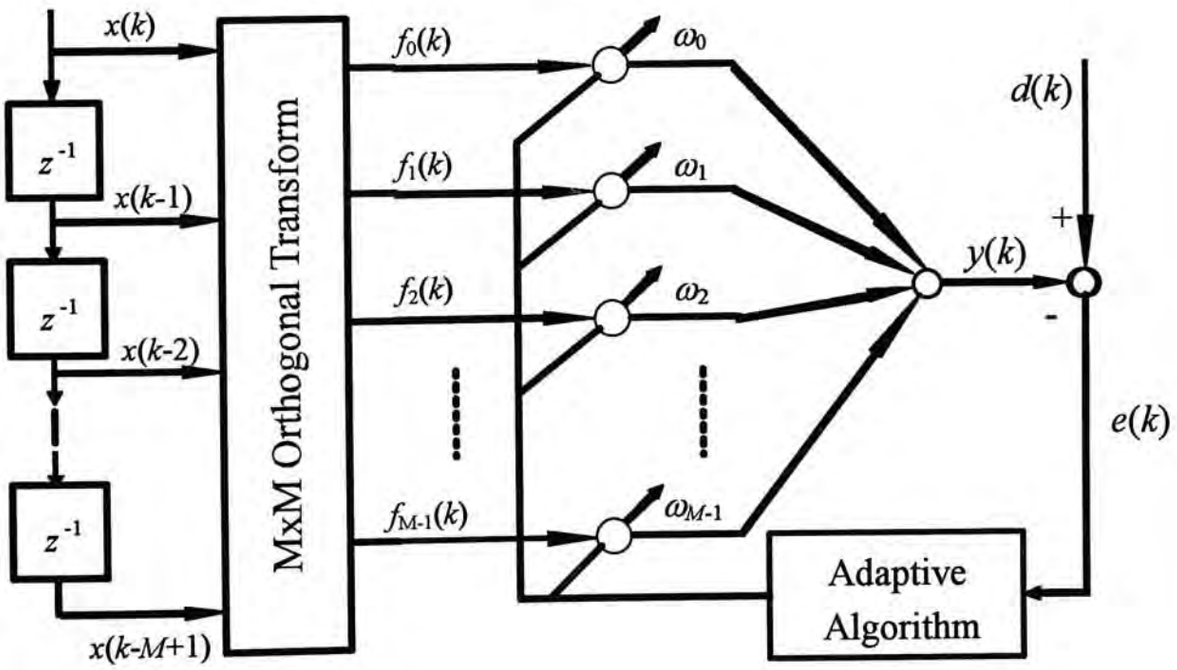


Fig. 4.2 Realization of a transform domain adaptive filter.

Transform	Transform matrix components	Factor K
DFT	$v_{i,j} = e^{-j2\pi(i-1)(j-1)/M}$	M
FKLT	$v_{i,j} = \sin\dfrac{ij\pi}{M+1}$	$\dfrac{M+1}{2}$
SCT	$v_{i,j} = \begin{cases} \dfrac{1}{\sqrt{2}}u(j)\cos\dfrac{ij\pi}{M-1} & i = 0, M-1 \\ u(j)\cos\dfrac{ij\pi}{M-1} & \text{elsewhere} \end{cases}$	$\dfrac{M-1}{2}$
DST	$v_{i,j} = \begin{cases} \sin\dfrac{i(2j+1)\pi}{2M} & \text{elsewhere} \\ \dfrac{(-1)^{j-1}}{\sqrt{2}} & i = M-1 \end{cases}$	$\dfrac{M}{2}$
DCT	$v_{i,j} = \begin{cases} \dfrac{1}{\sqrt{2}} & i = 0 \\ \cos\dfrac{\pi(2j+1)i}{2M} & \text{elsewhere} \end{cases}$	$\dfrac{M}{2}$
DHT	$v_{i,j} = \text{cas}(2\pi ij / M)$ $= \cos(2\pi ij / M) + \sin(2\pi ij / M)$	M
DWT	$\text{Wal}(i,j) = \prod_{r=0}^{L-1}(-1)^{i_{L-1-r}(j_r + j_{r+1})}$	M

Table 4.1. Property of some discrete transforms.

4.3.2 Properties of Transform Domain Adaptive Filters

According to (4.23), the error signal, $e(k)$, of a transform domain adaptive filter (TRAF) is given by the difference between the desired response $d(k)$ and the filter output $y(k)$, that is

$$\begin{aligned} e(k) &= d(k) - y(k) \\ &= d(k) - \boldsymbol{\omega}_k^T \boldsymbol{f}_k \end{aligned} \tag{4.24}$$

where the MSE, ε^t, is defined as

$$\varepsilon^t = E[e^2(k)] \tag{4.25}$$

and superscript t denotes the variable which is in transform domain. For gradient-based adaptive algorithms, the MSE is minimized by updating $\boldsymbol{\omega}_k$ according to

$$\boldsymbol{\omega}_{k+1} = \boldsymbol{\omega}_k - \eta(\nabla \varepsilon^t)_k \tag{4.26}$$

where η is a convergence factor for the transform domain parameter, and $(\nabla \varepsilon^t)_k$ is the gradient vector at the kth iteration. Define

$$\xi \triangleq E[d^2(k)] \tag{4.27}$$

$$\Phi_{df} \triangleq E[d(k)\boldsymbol{f}_k] \qquad \text{and} \qquad \Phi_f \triangleq E[\boldsymbol{f}_k \boldsymbol{f}_k^T] \tag{4.28}$$

It has been shown by Lee and Un [Lee 1986] that the optimal weight vector, $\boldsymbol{\omega}^*$ and the minimum MSE in the transform domain are related to their counterparts in the time domain by the following equations, that is

$$\boldsymbol{\omega}^* = \Phi_f^{-1}\Phi_{df} = (\boldsymbol{V}^T)^{-1}\boldsymbol{w}^* \tag{4.29}$$

and

$$\varepsilon_{\min}^{t} = \xi - \Phi_{df}^{T}\Phi_{f}^{-1}\Phi_{df} = \varepsilon_{min} \tag{4.30}$$

where w_k, μ and ε denote the weight vector, convergence factor and the MSE in time domain adaptive filter (TDAF), respectively. It was also shown in [Lee 1986] that if $\eta = \mu / K$ is used in the transform domain LMS algorithm, then the steady-state MSE of TDAF and TRAF will become identical. However, the convergence speed of TRAF is significantly faster because the eigenvalues of TRAF is approximately diagonalized by an orthogonal transform.

4.4 Transform Domain Split-Path LMS Adaptive Predictor

It has been reported that splitting technique can be applied to some classical algorithms [Delsarte 1987], such as the Schur algorithm [Roux 1977], the lattice algorithm [Itakura 1971] and the normalized lattice algorithm [Gray 1975] to achieve better adaptation performance. In Chapter 2, we have discussed the properties and the advantages of the split structure. The successful applications of split structure can also be found in autoregressive (AR) modeling [Ho 1992a], system identification [Wan 1995c] and linear prediction [Wan 1992].

Now, we shall make an exhaustive investigation of the split structure in transform domain adaptation. Fig. 4.3. shows the schematic block diagram of a transform domain split-path adaptive predictor which consists of two adaptive subsystems connected in parallel [Ho 1992b].

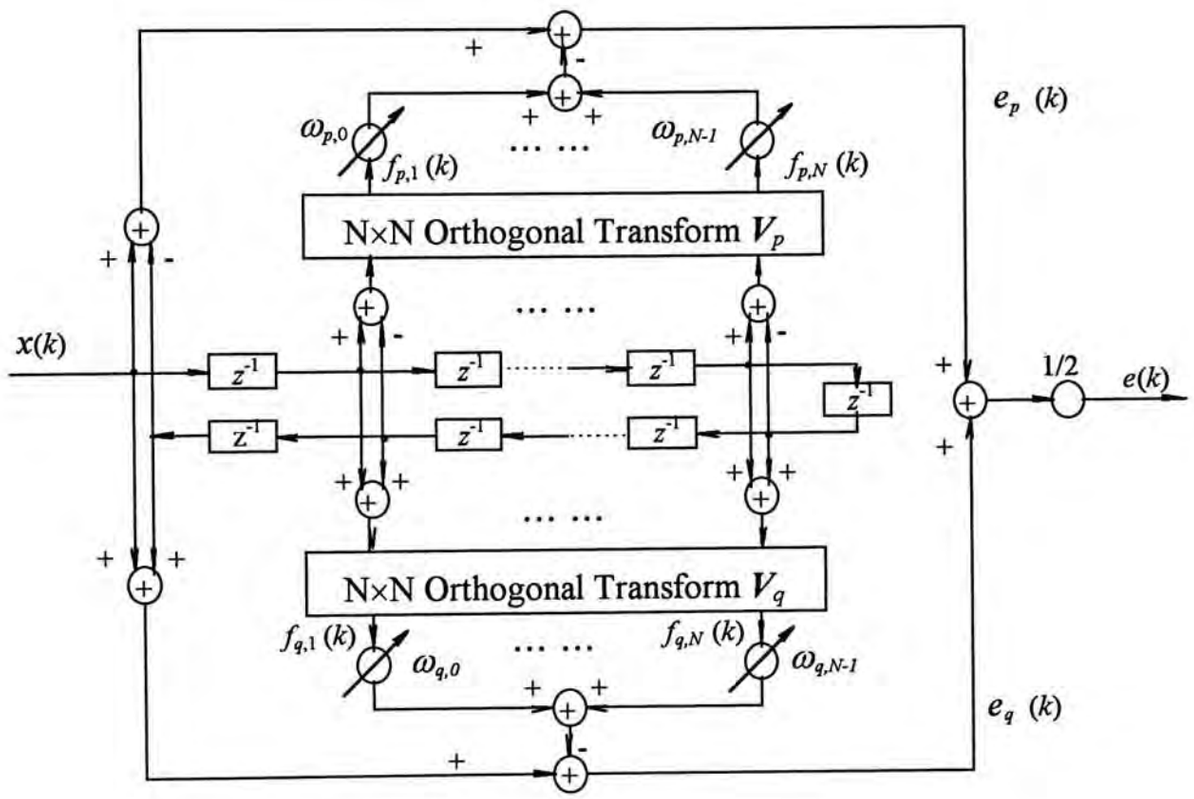


Fig. 4.3. Realization of a transform domain split-path adaptive predictor.

The two modified regression input vectors, $\boldsymbol{s}_{p,k}$ and $\boldsymbol{s}_{q,k}$, are of the form

$$\boldsymbol{s}_{p,k} = [x(k-1) - x(k-2N),\ \cdots,\ x(k-N) - x(k-N-1)]^T \tag{4.31}$$

and

$$\boldsymbol{s}_{q,k} = [x(k-1) + x(k-2N),\ \cdots,\ x(k-N) + x(k-N-1)]^T \tag{4.32}$$

and they are first transformed into $\boldsymbol{f}_{p,k}$ and $\boldsymbol{f}_{q,k}$, where

$$\boldsymbol{f}_{p,k} = \boldsymbol{V}_p\, \boldsymbol{s}_{p,k} = [f_{p,1}(k), f_{p,2}(k),\ \cdots,\ f_{p,N}(k)]^T \tag{4.33}$$

and

$$\boldsymbol{f}_{q,k} = \boldsymbol{V}_q\, \boldsymbol{s}_{q,k} = [f_{q,1}(k), f_{q,2}(k),\ \cdots,\ f_{q,N}(k)]^T \tag{4.34}$$

and $\boldsymbol{V}_p$ and $\boldsymbol{V}_q$ are two $N \times N$ orthonormal transformation matrices.

The weight vectors of two branches are

$$\boldsymbol{\omega}_{p,k} = [\omega_{p,0}(k),\ \omega_{p,1}(k),\ \cdots,\ \omega_{p,N-1}(k)]^T \tag{4.35}$$

and

$$\boldsymbol{\omega}_{q,k} = [\omega_{q,0}(k),\ \omega_{q,1}(k),\ \cdots,\ \omega_{q,N-1}(k)]^T \tag{4.36}$$

The branch outputs, $y_p(k)$ and $y_q(k)$, are the inner products of the respective input and filter weights. The outputs errors of the two filter path can be obtained from

$$\begin{aligned} e_p(k) &= \{x(k) - x(k-M-1)\} - y_p(k) \\ &= \{x(k) - x(k-M-1)\} - \boldsymbol{f}_{p,k}^T \boldsymbol{\omega}_{p,k} \end{aligned} \tag{4.37}$$

and

$$\begin{aligned} e_q(k) &= \{x(k) + x(k-M-1)\} - y_q(k) \\ &= \{x(k) + x(k-M-1)\} - \boldsymbol{f}_{q,k}^T \boldsymbol{\omega}_{q,k} \end{aligned} \tag{4.38}$$

If the LMS algorithm is used, the weight update equations in transform domain can be formulated as [Narayan 1983]

$$\begin{aligned} \boldsymbol{\omega}_{p,k+1} &= \boldsymbol{\omega}_{p,k} + 2\eta_p \Lambda_p^{-2}\, e_p(k)\boldsymbol{f}_{p,k} \\ \Lambda_p^2 &= \mathrm{diag}\{\sigma_{p,1}^2,\ \cdots,\ \sigma_{p,N}^2\} \end{aligned} \tag{4.39}$$

and

$$\begin{aligned} \boldsymbol{\omega}_{q,k+1} &= \boldsymbol{\omega}_{q,k} + 2\eta_q \Lambda_q^{-2}\, e_q(k)\boldsymbol{f}_{q,k} \\ \Lambda_q^2 &= \mathrm{diag}\{\sigma_{q,1}^2,\ \cdots,\ \sigma_{q,N}^2\} \end{aligned} \tag{4.40}$$

where η_p and η_q are the step sizes for branch p and q, respectively. The parameters $\sigma^2_{p,i}$ and $\sigma^2_{q,i}$ are power estimates that can be computed by taking exponentially weighted average of the square of N previous samples as shown below,

$$\sigma_{p,i}^2(k+1) = \rho\sigma_{p,i}^2(k) + (1-\rho)f_{p,i}^2(k), \qquad i = 1,\ 2,\ \cdots,\ N \tag{4.41}$$

$$\sigma_{q,i}^2(k+1) = \rho\sigma_{q,i}^2(k) + (1-\rho)f_{q,i}^2(k), \qquad i = 1,\ 2,\ \cdots,\ N \tag{4.42}$$

We next consider the steady state solution of the new adaptive system. Taking statistical expectation of (4.39) yields

$$E[\boldsymbol{\omega}_{p,k+1}] = E[\boldsymbol{\omega}_{p,k}] + 2\eta_p \Lambda_p^{-2} E[e_p(k)\boldsymbol{f}_{p,k}] \tag{4.43}$$

As long as steady state is maintained,

$$E[\boldsymbol{\omega}_{p,k+1}] = E[\boldsymbol{\omega}_{p,k}] \tag{4.44}$$

Since ρ is chosen to be sufficiently large, $\sigma^2_{p,i}$ and $\sigma^2_{q,i}$ change very slowly. Then $\Lambda_p{}^2$ can be assumed to be a nonsingular constant. Put (4.44) in (4.43), the optimal solution $\boldsymbol{\omega}_p^*$ can be obtained by solving

$$E[e_p(k)\boldsymbol{f}_{p,k}] = 0 \tag{4.45}$$

After substituting (4.33) and (4.37) into (4.45), we get

$$E\left[\{x(k) - x(k-M-1) - \boldsymbol{s}_{p,k}^T \boldsymbol{V}_p^T \boldsymbol{\omega}_p^*\}\boldsymbol{V}_p \boldsymbol{s}_{p,k}\right] = 0 \tag{4.46}$$

That is

$$E[\boldsymbol{s}_{p,k}\boldsymbol{s}_{p,k}^T]\,\boldsymbol{V}_p^T\boldsymbol{\omega}_p^* = E[(x(k)-x(k-M-1))\,\boldsymbol{s}_{p,k}] \tag{4.47}$$

It has been shown by Ching and Ho [Ching 1991] that the optimal solution of a time domain split-path adaptive predictor is,

$$\boldsymbol{p}^* = E[\boldsymbol{s}_{p,k}\boldsymbol{s}_{p,k}^T]^{-1}\,E[(x(k)-x(k-M-1))\,\boldsymbol{s}_{p,k}] \tag{4.48}$$

therefore, $\boldsymbol{\omega}_p^*$ can be related to $\boldsymbol{p}^*$ by,

$$\boldsymbol{\omega}_p^* = \boldsymbol{V}_p\,\boldsymbol{p}^* \tag{4.49}$$

Similarly, we have

$$\boldsymbol{\omega}_q^* = \boldsymbol{V}_q\,\boldsymbol{q}^* \tag{4.50}$$

If η_p is chosen to be small, then $\boldsymbol{f}_{p,k}$ and $\boldsymbol{\omega}_{p,k}$ can be regarded as independent to each other, and in this case, the gradient estimate in (4.39) can be expressed as

$$E[e_p(k)\boldsymbol{f}_{p,k}] = -E[\boldsymbol{f}_{p,k}\boldsymbol{f}_{p,k}^T]E[\Delta\boldsymbol{\omega}_{p,k}] \tag{4.51}$$

where $\Delta\boldsymbol{\omega}_{p,k} = \boldsymbol{\omega}_{p,k} - \boldsymbol{\omega}^*$ is the parameter estimation error. Hence (4.39) can be simplified to

$$E[\Delta\boldsymbol{\omega}_{p,k+1}] = (\mathrm{I}-2\eta_p\Lambda_p^{-2}\,E[\boldsymbol{f}_{p,k}\boldsymbol{f}_{p,k}^T])\,E[\Delta\boldsymbol{\omega}_{p,k}] \tag{4.52}$$

The adaptation speed and the stability range of η_p is obviously governed by the eigenvalue of $\Lambda_p^{-2}E[\boldsymbol{f}_{p,k}\,\boldsymbol{f}_{p,k}^T]$. Due to the decorrelation ability of transformation [Narayan 1983], the eigenvalue spread for branch p will be much smaller than $E[\boldsymbol{s}_{p,k}\,\boldsymbol{s}_{p,k}^T]$, which is the autocorrelation matrix in the time domain split-path system. That is, $\Lambda_p^{-2}E[\boldsymbol{f}_{p,k}\,\boldsymbol{f}_{p,k}^T]$ is approximately diagonal. As a result, a faster convergence rate is obtained. Similar argument also applies to the branch q.

Combining (4.37) and (4.38), the overall system error can be expressed as

$$e(k)=\frac{1}{2}[e_p(k)+e_q(k)]=\frac{1}{2}[x(k)-(\boldsymbol{f}_{p,k}^T\boldsymbol{\omega}_{p,k}+\boldsymbol{f}_{q,k}^T\boldsymbol{\omega}_{q,k})] \tag{4.53}$$

Although $x(k\text{-}M\text{-}1)$ is found required in computing $e_p(k)$ and $e_q(k)$, it is actually not necessary for computing the overall output error $e(k)$. Because $e_p(k)$ and $e_q(k)$ only correlate with their own path parameters, thus we have

$$E[e(k)\boldsymbol{f}_{p,k}]=E[e_p(k)\boldsymbol{f}_{p,k}] \quad \text{and} \quad E[e(k)\boldsymbol{f}_{q,k}]=E[e_q(k)\boldsymbol{f}_{q,k}] \tag{4.54}$$

This implies that only identical $e(k)$ can be used in the gradient estimations for the pair of parameter updating equations, that is (4.39) and (4.40) can be replaced by

$$\boldsymbol{\omega}_{p,k+1}=\boldsymbol{\omega}_{p,k}+2\eta_p\Lambda_p^{-2}\,e(k)\boldsymbol{f}_{p,k} \tag{4.55}$$

$$\boldsymbol{\omega}_{q,k+1}=\boldsymbol{\omega}_{q,k}+2\eta_q\Lambda_q^{-2}\,e(k)\boldsymbol{f}_{q,k} \tag{5.56}$$

The constant 1/2 in (4.54) is absorbed in the convergence factors.

Hence, we can extend the structure of transform domain split-path adaptive filter (TRSPAF) in a more general form as shown in Fig. 4.4. Assuming the original M-weights time domain adaptive filter has transfer function $W(z)$, where M is even and $N=M/2$. The input vectors, $\boldsymbol{f}_{p,k}$ and $\boldsymbol{f}_{q,k}$, for the two branches of the new TRSPAF system can be obtained from

$$\begin{bmatrix}\boldsymbol{f}_{p,k}\\ \boldsymbol{f}_{q,k}\end{bmatrix}=\boldsymbol{U}_1\boldsymbol{f}_k=\boldsymbol{U}_1\boldsymbol{V}\boldsymbol{x}_k \tag{4.57}$$

where $\boldsymbol{U}_1$ is same as that defined in Chapter 2. In (4.57)

$$\boldsymbol{f}_{p,k}=\begin{bmatrix}\boldsymbol{I}_1 & -\boldsymbol{J}_1\end{bmatrix}\boldsymbol{V}\boldsymbol{x}_k \tag{4.58}$$

$$\boldsymbol{f}_{q,k}=\begin{bmatrix}\boldsymbol{I}_1 & \boldsymbol{J}_1\end{bmatrix}\boldsymbol{V}\boldsymbol{x}_k \tag{4.59}$$

where $\boldsymbol{I}_1$ is an $N\times N$ identity matrix and $\boldsymbol{J}_1$ is an anti-diagonal identity matrix of the same size. Comparing (4.58) and (4.59) with (4.33) and (4.34), we have noticed that

the splitting procedure can be implemented in two ways. One is to separate the signal in time domain, and then make the transformation separately for the two branches individually, which is shown in Fig. 4.4. Another approach is to transform the input signal first, and then split the transform domain signal into two and feed them into a pair of adaptive subfilters both with $M/2$ parameters. This method is shown in Fig. 4.5. It is trivial that irrespective to whichever method is used, the results are identical. The relationship between the parameter vectors can alternatively be expressed as

$$\boldsymbol{\omega}_k^T = \begin{bmatrix} \boldsymbol{\omega}_{p,k}^T & \boldsymbol{\omega}_{q,k}^T \end{bmatrix} \boldsymbol{U}_1 \tag{4.60}$$

The overall system output error is obtained by subtracting the sum of the branch outputs, $y_p(k) = \boldsymbol{f}_{p,k}^T \boldsymbol{\omega}_{p,k}$ and $y_q(k) = \boldsymbol{f}_{q,k}^T \boldsymbol{\omega}_{q,k}$, from the desired response,

$$e(k) = d(k) - (\boldsymbol{f}_{p,k}^T \boldsymbol{\omega}_{p,k} + \boldsymbol{f}_{q,k}^T \boldsymbol{\omega}_{q,k}) \tag{4.61}$$

Squaring (4.61) and taking expectation yields the MSE of TRSPAF,

$$\varepsilon^{sp} = \xi - 2\boldsymbol{\omega}_{p,k}^T \Phi_{dp} - 2\boldsymbol{\omega}_{q,k}^T \Phi_{dq} + (\boldsymbol{\omega}_{p,k}^T \Phi_p \boldsymbol{\omega}_{p,k} + \boldsymbol{\omega}_{q,k}^T \Phi_q \boldsymbol{\omega}_{q,k}) + 2\boldsymbol{\omega}_{pk}^T \Phi_{pq} \boldsymbol{\omega}_{q,k} \tag{4.62}$$

where superscript *sp* denotes the split-path structure, and

$$\Phi_p \triangleq E[\boldsymbol{f}_{p,k} \boldsymbol{f}_{p,k}^T] \qquad \text{and} \qquad \Phi_q \triangleq E[\boldsymbol{f}_{q,k} \boldsymbol{f}_{q,k}^T] \tag{4.63}$$

are the autocorrelation matrices of $\boldsymbol{f}_{p,k}$ and $\boldsymbol{f}_{q,k}$, and

$$\Phi_{dp} \triangleq E[d(k)\boldsymbol{f}_{p,k}^T], \quad \Phi_{dq} \triangleq E[d(k)\boldsymbol{f}_{q,k}^T] \quad \text{and} \quad \Phi_{pq} \triangleq E[\boldsymbol{f}_{p,k} \boldsymbol{f}_{q,k}^T] \tag{4.64}$$

are the cross-correlation matrices. From equations (4.58), (4.59),

$$\begin{aligned} \Phi_{pq} &= E\left(\begin{bmatrix} \boldsymbol{I}_1 & -\boldsymbol{J}_1 \end{bmatrix} \boldsymbol{V} \boldsymbol{x}_k \boldsymbol{x}_k^T \boldsymbol{V}^T \begin{bmatrix} \boldsymbol{I}_1^T \\ \boldsymbol{J}_1^T \end{bmatrix} \right) \\ &= \begin{bmatrix} \boldsymbol{I}_1 & -\boldsymbol{J}_1 \end{bmatrix} \Phi_f \begin{bmatrix} \boldsymbol{I}_1 \\ \boldsymbol{J}_1 \end{bmatrix} \end{aligned} \tag{4.65}$$

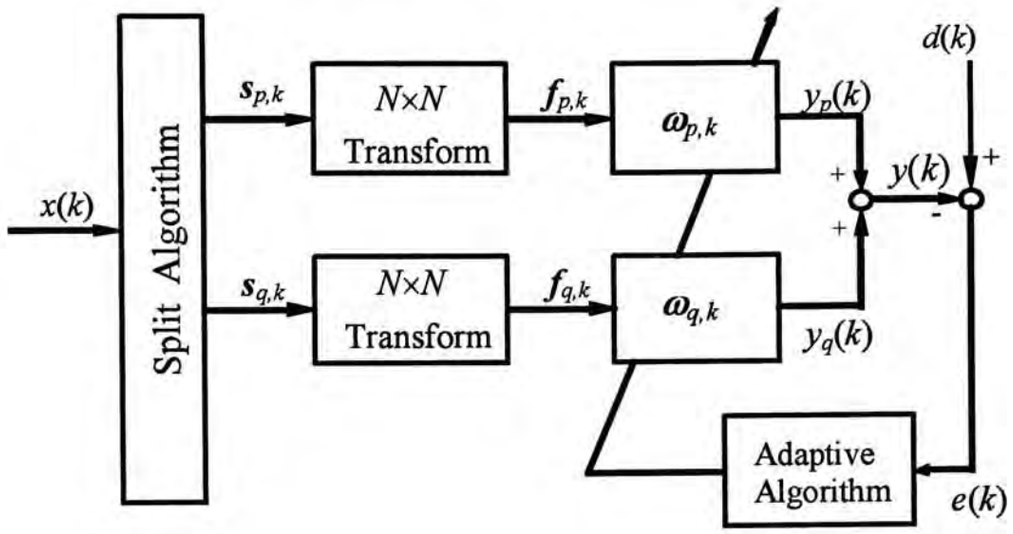


Fig. 4.4 Transform domain split-path adaptive system - method 1.

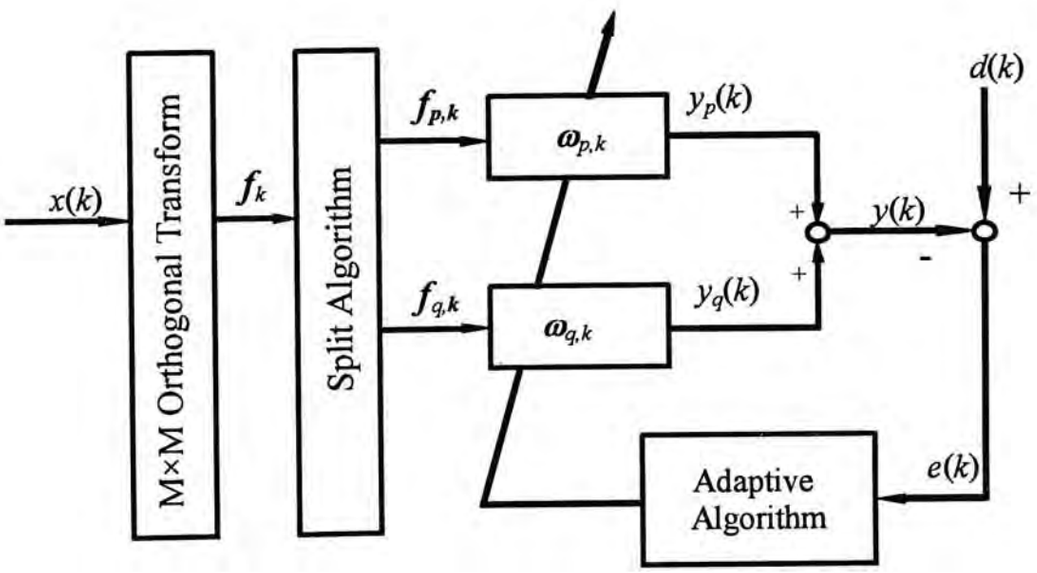


Fig. 4.5 Transform domain split-path adaptive system - method 2.

The transform domain autocorrelation matrix Φ_f is still a symmetric Toeplitz matrix and can be divided into two Toeplitz submatrices, Φ_1 and Φ_2, which can be expressed as

$$\Phi_f = \begin{bmatrix} \Phi_1 & \Phi_2^T \\ \Phi_2 & \Phi_1 \end{bmatrix} \tag{4.66}$$

It is easy to show that

$$\boldsymbol{J}_1\Phi_2 = \Phi_2^T\boldsymbol{J}_1 \qquad \text{and} \qquad \boldsymbol{J}_1\Phi_1\boldsymbol{J}_1 = \Phi_1^T \tag{4.67}$$

thus, we have

$$\begin{aligned} \Phi_{pq} &= \Phi_1 - \boldsymbol{J}_1\Phi_2 + \Phi_2^T\boldsymbol{J}_1 - \boldsymbol{J}_1\Phi_1\boldsymbol{J}_1 \\ &= 0 \end{aligned} \tag{4.68}$$

Hence (4.62) can be simplified to

$$\varepsilon^{sp} = \xi - 2\boldsymbol{\omega}_{p,k}^T\Phi_{dp} - 2\boldsymbol{\omega}_{q,k}^T\Phi_{dq} + (\boldsymbol{\omega}_{p,k}^T\Phi_p\boldsymbol{\omega}_{p,k} + \boldsymbol{\omega}_{q,k}^T\Phi_q\boldsymbol{\omega}_{q,k}) \tag{4.69}$$

It is noticed that the expressions for the split-path adaptive algorithm in transform domain coincide with those in time domain, therefore, the analysis method that is commonly used in time domain can be adopted as well. The decoupled parameters are now being adjusted simultaneously and independently by minimizing ε^{sp} using the gradient-based updating algorithm,

$$\boldsymbol{\omega}_{p,k+1} = \boldsymbol{\omega}_{p,k} - \eta_p(\nabla_p\varepsilon^{sp})_k \tag{4.70}$$

and

$$\boldsymbol{\omega}_{q,k+1} = \boldsymbol{\omega}_{q,k} - \eta_q(\nabla_q\varepsilon^{sp})_k \tag{4.71}$$

where η_p and η_q are the convergence factors for $\boldsymbol{\omega}_{p,k}$ and $\boldsymbol{\omega}_{q,k}$, which can be assigned with different values. While ∇_p and ∇_q represent the gradient operators corresponding to the vector $\boldsymbol{\omega}_{p,k}$ and $\boldsymbol{\omega}_{q,k}$, respectively, that is

$$(\nabla \varepsilon^{sp})_k = \begin{bmatrix} \nabla_p \\ \nabla_q \end{bmatrix} (\varepsilon^{sp})_k \tag{4.72}$$

4.5 Performance Analysis of the Transform Domain Split-Path Adaptive Algorithm

The MSE and the convergence speed are two important performance measures and in general, they both depend on the choices of the convergence factor. In this section, we shall derive the optimum Wiener solution and the steady state MSE for transform domain split-path adaptive filtering (TRSPAF).

4.5.1 Optimum Wiener Solution

Since $\boldsymbol{f}_{p,k}$ and $\boldsymbol{f}_{q,k}$ are independent to each other, the optimum weight vectors $\boldsymbol{\omega}_p^*$ and $\boldsymbol{\omega}_q^*$ can be obtained by substituting (4.69) into (4.72) and equating the gradient vector to zero, thus

$$\begin{bmatrix} (\nabla_p \varepsilon^{sp})_k \\ (\nabla_q \varepsilon^{sp})_k \end{bmatrix} = 2 \begin{bmatrix} -\Phi_{dp} + \Phi_p \boldsymbol{\omega}_{p,k} \\ -\Phi_{dq} + \Phi_q \boldsymbol{\omega}_{q,k} \end{bmatrix} = 0 \tag{4.73}$$

which gives

$$\begin{bmatrix} \boldsymbol{\omega}_p^* \\ \boldsymbol{\omega}_q^* \end{bmatrix} = \begin{bmatrix} \Phi_p^{-1} & 0 \\ 0 & \Phi_q^{-1} \end{bmatrix} \begin{bmatrix} \Phi_{dp} \\ \Phi_{dq} \end{bmatrix} \tag{4.74}$$

Putting (4.74) into (4.69) and using the properties $(\Phi_p^{-1})^T = \Phi_p^{-1}$ and $(\Phi_q^{-1})^T = \Phi_q^{-1}$, we obtain the minimum MSE for TRSPAF, which is given by

$$\varepsilon_{min}^{sp} = \xi - \Phi_{dp}^T \Phi_p^{-1} \Phi_{dp} - \Phi_{dq}^T \Phi_q^{-1} \Phi_{dq} \tag{4.75}$$

According to the definitions in (4.27) and (4.64), we have

$$\begin{bmatrix} \Phi_{dp}^T & \Phi_{dq}^T \end{bmatrix} = \Phi_{df}^T \boldsymbol{U}_1^T \tag{4.76}$$

and

$$\begin{aligned} \begin{bmatrix} \Phi_p & 0 \\ 0 & \Phi_q \end{bmatrix} &= E\left(\begin{bmatrix} \boldsymbol{f}_{p,k} \\ \boldsymbol{f}_{q,k} \end{bmatrix} \begin{bmatrix} \boldsymbol{f}_{p,k}^T & \boldsymbol{f}_{q,k}^T \end{bmatrix} \right) \\ &= \boldsymbol{U}_1 \Phi_f \boldsymbol{U}_1^T = \boldsymbol{U}_1 (2\Phi_f)\, \boldsymbol{U}_1^{-1} \end{aligned} \tag{4.77}$$

hence, equations (4.74) and (4.75) can be further simplified to

$$\begin{bmatrix} \boldsymbol{\omega}_p^* \\ \boldsymbol{\omega}_q^* \end{bmatrix} = \frac{1}{2} \boldsymbol{U}_1 \Phi_f^{-1} \boldsymbol{U}_1^{-1} \boldsymbol{U}_1 \Phi_{df} = \frac{1}{2} \boldsymbol{U}_1 \Phi_f^{-1} \Phi_{df} \tag{4.78}$$

and

$$\begin{aligned} \varepsilon_{min}^{sp} &= \xi - \begin{bmatrix} \Phi_{dp}^T & \Phi_{dq}^T \end{bmatrix} \begin{bmatrix} \Phi_p^{-1} & 0 \\ 0 & \Phi_q^{-1} \end{bmatrix} \begin{bmatrix} \Phi_{dp} \\ \Phi_{dq} \end{bmatrix} \\ &= \xi - \Phi_{df}^T \boldsymbol{U}_1^T \boldsymbol{U}_1^{-T} \Phi_f^{-1} \boldsymbol{U}_1^{-1} \boldsymbol{U}_1 \Phi_{df} = \xi - \Phi_{df} \Phi_f^{-1} \Phi_{df} \end{aligned} \tag{4.79}$$

respectively. In deriving (4.77), the orthogonal property of $\boldsymbol{U}_1^T \boldsymbol{U}_1 = 2\boldsymbol{I}$ is used. Combining (4.78) with (4.28) as well as (4.79) with (4.29), we have

$$\begin{bmatrix} \boldsymbol{\omega}_p^* \\ \boldsymbol{\omega}_q^* \end{bmatrix} = \frac{1}{2} \boldsymbol{U}_1 \boldsymbol{\omega}^* = \frac{1}{2} \boldsymbol{U}_1 (\boldsymbol{V}^T)^{-1} \boldsymbol{w}^* \tag{4.80}$$

$$\varepsilon_{min}^{sp} = \varepsilon_{min} = \varepsilon_{min}^{t} \tag{4.81}$$

This indicates that the minimum MSE for TRSPAF, TRAF and TDAF are all identical. Furthermore, there is a unique relationship, *viz.* (4.80), between the optimal weight vectors of these three different adaptive methods.

4.5.2 Steady State MSE and Convergence Speed

In our study, we again apply the well known LMS algorithm to adjust the filter parameters. The updating equation for TRAF is

$$\boldsymbol{\omega}_{k+1} = \boldsymbol{\omega}_k + 2\eta \boldsymbol{f}_k e(k) \tag{4.82}$$

whereas the updating equations for the TRSPAF are of the form

$$\boldsymbol{\omega}_{p,k+1} = \boldsymbol{\omega}_{p,k} + 2\eta_p \boldsymbol{f}_{p,k} e(k) \tag{4.83}$$

$$\boldsymbol{\omega}_{q,k+1} = \boldsymbol{\omega}_{q,k} + 2\eta_q \boldsymbol{f}_{q,k} e(k) \tag{4.84}$$

It is easy to show that the sufficient conditions for convergence are

$$0 < \eta < \frac{1}{tr(\Phi_f)} \qquad \text{for TRAF} \tag{4.85}$$

and

$$0 < \eta_p < \frac{1}{tr(\Phi_p)}, \qquad 0 < \eta_q < \frac{1}{tr(\Phi_q)} \qquad \text{for TRSPAF} \tag{4.86}$$

where tr(.) denotes the trace operator of the corresponding matrices. Since $tr(\Phi_f)=[tr(\Phi_p)+tr(\Phi_q)]/2$, therefore, a larger stability range is available for η_p and η_q. When steady state is reached, the excess MSE of TRAF equals

$$\varepsilon_\Delta = \eta\, tr(\Phi_f)\varepsilon_{min} \tag{4.87}$$

while the excess MSE of TRSPAF is given by

$$\varepsilon_\Delta^{sp} = [\eta_p\, tr(\Phi_p) + \eta_q\, tr(\Phi_q)]\, \varepsilon_{min} \tag{4.88}$$

Contrasting (4.87) with (4.88), we note that the same steady state excess MSE will be obtained in both cases if the following condition is satisfied

$$\eta\, tr(\Phi_f) = [\eta_p\, tr(\Phi_p) + \eta_q\, tr(\Phi_q)] \tag{4.89}$$

One disadvantage of the LMS algorithm is that the convergence speed is restricted by the eigenvalue spread of the input covariance matrix. From (4.77), it is seen that the eigenvalues of Φ_p and Φ_q can be computed by partitioning the

eigenvalues of Φ_f with a scaling factor of 2. Let γ be the eigenvalue spread of Φ_f, the eigenvalue spreads of Φ_p and Φ_q, denoted by γ_p and γ_q, then the following inequalities are always satisfied

$$\gamma_p = \frac{\lambda_{p,max}}{\lambda_{p,min}} \leq \frac{\lambda_{max}}{\lambda_{min}} = \gamma \tag{4.90}$$

$$\gamma_q = \frac{\lambda_{q,max}}{\lambda_{q,min}} \leq \frac{\lambda_{max}}{\lambda_{min}} = \gamma \tag{4.91}$$

where λ_{max}, $\lambda_{p,max}$ and $\lambda_{q,max}$ represent the maximum, and λ_{min}, $\lambda_{p,min}$ and $\lambda_{q,min}$ represent the minimum eigenvalues of Φ_f, Φ_p and Φ_q, respectively. When introducing the power estimates, Λ_p^2 and Λ_q^2, as the self-orthogonal diagonal matrices, the parameter updating equations become (4.55) and (4.56). Hence, it is anticipated that the split-path adaptive structure can converge faster that the non-split configuration with the exception that when both maximum and minimum eigenvalues are distributed in the same path. In this unlikely event, the branch that includes λ_{max} and λ_{min} will reduce to a similar convergence rate as the original non-split structure, but the other branch still offers a faster convergence speed.

4.6 Computer Simulation Examples

In this section we shall discuss the convergence behaviour of the transform domain split-path adaptive system based on the results of computer simulation.

A speech like correlated input signal generated by an all pole filter with random input was to be identified first by the transform domain split-path predictor, then by

the transform domain TDL predictor and lastly, by the time domain split-path system. The pole locations were selected to be [Ho 1992b]

$$0.95\angle\pm30°,\ 0.9\angle\pm45°,\ 0.85\angle\pm80°\ \text{ and }\ 0.8\angle\pm120° \tag{4.92}$$

The transform being used was the DCT. In order to make fair comparison of different methods, the convergence factors in the three linear predictors were assigned to maintain a fixed level of misadjustment in all tests. Fig. 4.6 compares the convergence characteristics of the split-path system for time domain and transform domain adaptation. It is observed that adaptation in transform domain not only increases the adaptation speed but also reduces the final steady state mean square error substantially. Fig. 4.7 shows the learning characteristics of the split-path and TDL transform domain adaptive predictors. Again, the rate of convergence of the proposed system can be seen to be faster and MSE was reduced to a value of 0.05 at about 200 iterations which was roughly 300 iterations less than the other. In addition, temporal variation of the trajectory of the split-path model is a lot smaller, indicating that the gradient noise is lessened in the transient period. This is essential because the learning behaviour of system parameters is usually one of the major concerns in adaptive systems.

Extensive simulations using different transforms and input signals have also been performed and it is validated that the new split-path structure can outperform the TDL filter in terms of convergence speed, tracking characteristics as well as smoother trajectory. To illustrate this, the above experiment was repeated for the split-path and TDL models with DHT domain adaptation and their respective learning characteristics are depicted in Fig. 4.8 for comparison.

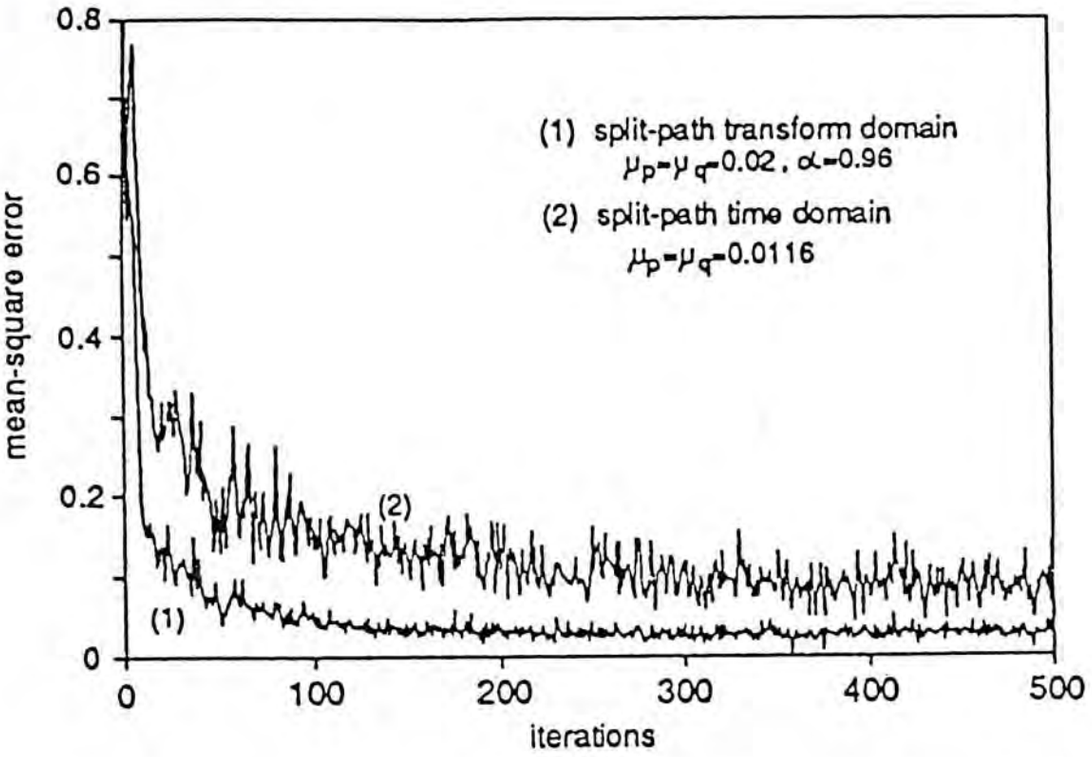


Fig. 4.6 Comparison of convergence speed of the split-path system for time domain and transform domain (DCT) adaptation.

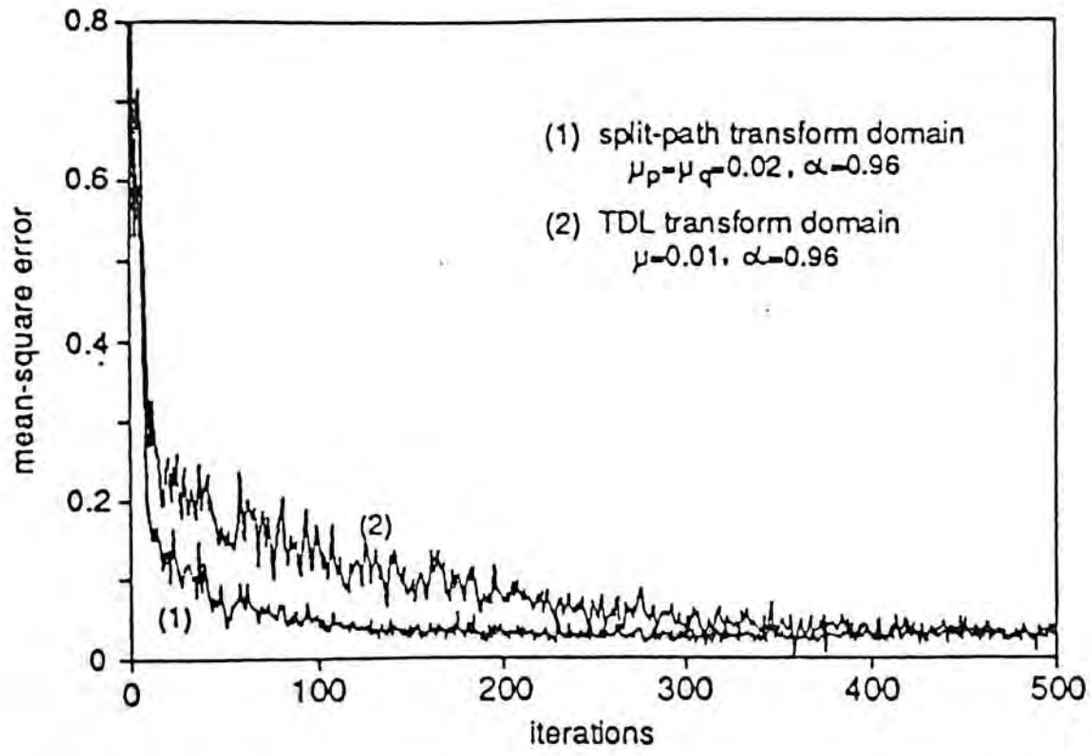


Fig. 4.7 Comparison of convergence speed between split-path and TDL transform domain (DCT) adaptive system.

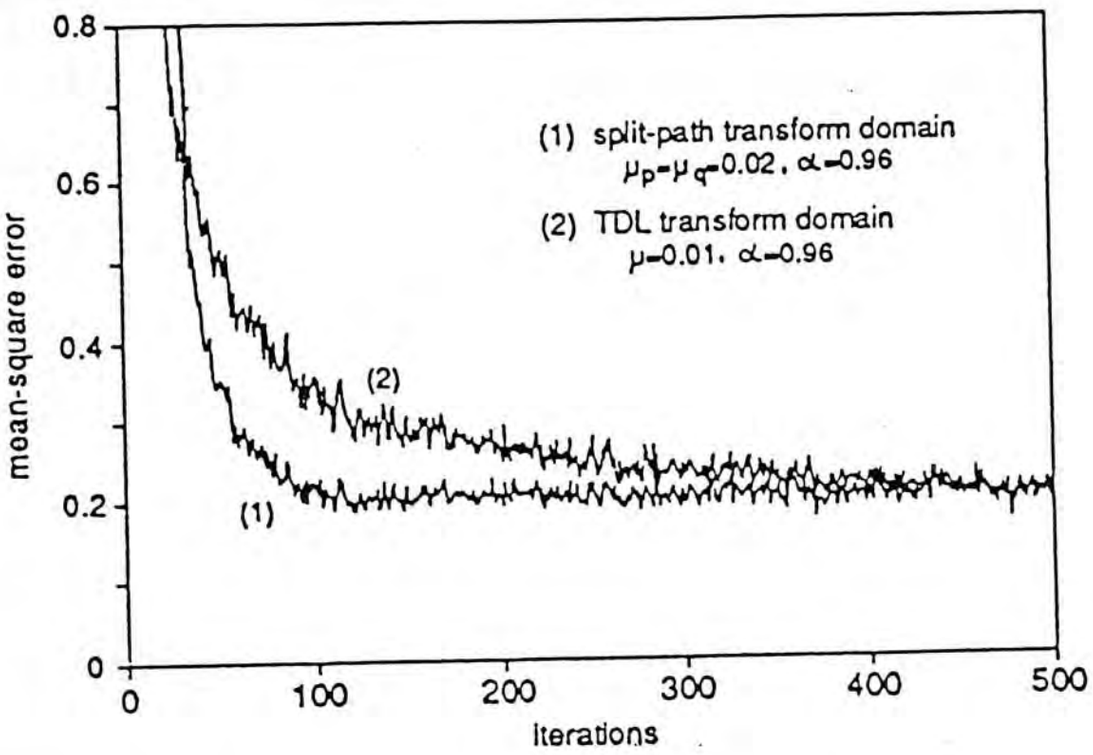


Fig. 4.8 Comparison of convergence speed between split-path and TDL transform domain (DHT) adaptive system.

4.7 Summary

First, some important transforms and their main characteristics are briefly introduced and compared to each other. Then, the application of transform domain adaptive technique to a linear predictor which is configured in a split-path structure is investigated. The properties and convergence performance of the novel adaptive system are analyzed rigorously. We have shown that the minimum MSE for TDAF, TRAF and TRSPAF are all identical while a unique conversion formula between optimal weight vectors of different configuration is established by means of discrete transform matrix and splitting matrix.

We have proved, in addition to the superior of transformation, that the eigenvalue spreads in the sub-branches of the split-path structure is always equal or less than the eigenvalue spread in a non-split configuration, such that convergence behaviour can be further enhanced. Simulation results by using different transformations have corroborated our theoretical analysis. It has also been found that the split-path adaptive algorithm in transform domain performs fairly similarly as in time domain, therefore, same basic analysis can also be employed for transform domain split-path adaptation.

Chapter 5 TRACKING OPTIMAL CONVERGENCE FACTOR FOR TRANSFORM DOMAIN SPLIT-PATH ADAPTIVE ALGORITHM

In Chapter 4, we have developed a new adaptive algorithm by applying the split-structure in transform domain adaptation. It has been shown that the split structure can effectively improve the statistical properties of the input signal and enhance the convergence dynamics of the adaptive system. However, in previous discussion, all step sizes are assumed to be constant, which somewhat restrict the system to achieve the best adaptation performance.

In this chapter, we shall devise a variable convergence factor technique for transform domain split-path adaptive filtering (TRSPAF) and analyze the adaptation characteristics of such systems. A pair of optimal convergence factors are derived for the general gradient-based split-path structure adaptive algorithm to achieve further convergence speed up and a self-adjusting method is developed facilitate for tracking of their optimal values. The method will be compared with the self-orthogonalizing method and simulation results are included to validate the theoretical analysis and to illustrate the advantages of the proposed algorithm [Wan 1994a] [Wan 1995b].

5.1 Introduction

In designing adaptive algorithms, whether in time domain or in transform domain, a proper choice of convergence factor is one of the most important considerations. In most cases, the selection of a right step size involves the tradeoff of misadjustment and speed of adaptation. Many works have been done to investigate the behaviour of the convergence factor and its effectiveness in improving the performance of adaptive algorithms. Generally speaking, a small step size will give a small misadjustment but a longer convergence time constant. Recently, many researchers have studied the possibility of using variable convergence factor to improve the adaptation performance, and quite a number of literatures have been published to this end [Yassa 1987].

Harris, *et al* [Harris 1986] has proposed a variable step (VS) adaptive filter algorithm, in which an extra feedback matrix has been introduced to modify the parameter updating equation as follows,

$$\boldsymbol{w}_{k+1} = \boldsymbol{w}_k + 2e(k)\boldsymbol{M}_k\boldsymbol{x}_k \tag{5.1}$$

where $\boldsymbol{M}_k$ is a diagonal matrix. The ith element, $\mu_i(k)$, of $\boldsymbol{M}_k$ is actually a time variable convergence factor associated with the ith parameter. Each component of $\boldsymbol{M}_k$ is changed according to the gradient estimation $e(k)\boldsymbol{x}_k$ and is allowed to vary between the fixed values μ_{max} and μ_{min}. The algorithm improves the convergence characteristics by providing a more accurate directional estimate in locating the minimum of the MSE surface.

Another method takes a new convergence factor, which is varying according to [Kwong 1992],

$$\mu(k+1)=a\mu(k)+be^2(k) \tag{5.2}$$

where $\mu(k)$ is controlled by the size of the prediction error $e^2(k)$, as well as by the parameters a and b. A pair of limits, μ_{max} and μ_{min}, are also used to ensure the stability. Intuitively, a large prediction error will cause the convergence factor to increase so as to provide faster tracking, while a small prediction error will result in a decrease in the convergence factor to yield smaller misadjustment.

Although, the above methods give considerable improvement to performance of the adaptive system, the complexity of the algorithm is also increased. In addition, for real computation, it is not easy to determine the value of μ_{max} and μ_{min}. Actually, it is somehow unreasonable to fix the value of μ_{max} and μ_{min} in an adaptive system, because they are closely connected with the input autocorrelation matrix $\boldsymbol{R}_x$. In fact, $\boldsymbol{R}_x$ is usually unknown particularly when the system is operating in a nonstationary environment. It has also been noticed that very few publications were found in discussing the applications of variable convergence factor for transform domain adaptation. In this chapter, the optimal convergence factor in transform domain split-path adaptive filtering will be discussed. We will derive an iterative optimal convergence factor tracking method for a split-path adaptive system. The proposed algorithm is capable of achieving a fast convergence speed but without introducing a larger misadjustment [Wan 1994a] [Wan 1995b].

5.2 The Optimal Convergence Factors of TRSPAF

In a gradient search adaptive system, the mean square error is gradually approaching to its steady state during adaptation. Hence, it is also a time variable estimate upon reception of the input data. Therefore, at time instant k, the MSE can be expressed as $\varepsilon^{sp}{}_k$, where the subscript k is used to emphasize the value at a certain iteration moment. The MSE of transform domain split-path adaptive filtering (TRSPAF) described by equation (4.69) can then be expressed more rigorously as follows,

$$\varepsilon_k^{sp} = \xi - 2\boldsymbol{\omega}_{p,k}^T \Phi_{dp} - 2\boldsymbol{\omega}_{q,k}^T \Phi_{dq} + (\boldsymbol{\omega}_{p,k}^T \Phi_p \boldsymbol{\omega}_{p,k} + \boldsymbol{\omega}_{q,k}^T \Phi_q \boldsymbol{\omega}_{q,k}) \tag{5.3}$$

or in the form of

$$\varepsilon_k^{sp} = \xi - 2\begin{bmatrix}\boldsymbol{\omega}_{p,k}^T & \boldsymbol{\omega}_{q,k}^T\end{bmatrix}\begin{bmatrix}\Phi_{dp} \\ \Phi_{dq}\end{bmatrix} + \begin{bmatrix}\boldsymbol{\omega}_{p,k}^T & \boldsymbol{\omega}_{q,k}^T\end{bmatrix}\begin{bmatrix}\Phi_p & 0 \\ 0 & \Phi_q\end{bmatrix}\begin{bmatrix}\boldsymbol{\omega}_{p,k} \\ \boldsymbol{\omega}_{q,k}\end{bmatrix} \tag{5.4}$$

While at the (k+1)th iteration, the MSE equals,

$$\varepsilon_{k+1}^{sp} = \xi - 2\boldsymbol{\omega}_{p,k+1}^T \Phi_{dp} - 2\boldsymbol{\omega}_{q,k+1}^T \Phi_{dq} + (\boldsymbol{\omega}_{p,k+1}^T \Phi_p \boldsymbol{\omega}_{p,k+1} + \boldsymbol{\omega}_{q,k+1}^T \Phi_q \boldsymbol{\omega}_{q,k+1}) \tag{5.5}$$

(5.5) can also be written as

$$\varepsilon_{k+1}^{sp} = \xi - 2\begin{bmatrix}\boldsymbol{\omega}_{p,k+1}^T & \boldsymbol{\omega}_{q,k+1}^T\end{bmatrix}\begin{bmatrix}\Phi_{dp} \\ \Phi_{dq}\end{bmatrix} + \begin{bmatrix}\boldsymbol{\omega}_{p,k+1}^T & \boldsymbol{\omega}_{q,k+1}^T\end{bmatrix}\begin{bmatrix}\Phi_p & 0 \\ 0 & \Phi_q\end{bmatrix}\begin{bmatrix}\boldsymbol{\omega}_{p,k+1} \\ \boldsymbol{\omega}_{q,k+1}\end{bmatrix} \tag{5.6}$$

In Chapter 4, we have indicated that

$$\begin{aligned}\begin{bmatrix}\boldsymbol{\omega}_{p,k+1} \\ \boldsymbol{\omega}_{q,k+1}\end{bmatrix} &= \begin{bmatrix}\boldsymbol{\omega}_{p,k} \\ \boldsymbol{\omega}_{q,k}\end{bmatrix} - \begin{bmatrix}\eta_p(\nabla_p \varepsilon^{sp})_k \\ \eta_q(\nabla_q \varepsilon^{sp})_k\end{bmatrix} \\ &= \begin{bmatrix}\boldsymbol{\omega}_{p,k} \\ \boldsymbol{\omega}_{q,k}\end{bmatrix} - \begin{bmatrix}\eta_p \boldsymbol{I} & 0 \\ 0 & \eta_q \boldsymbol{I}\end{bmatrix}(\nabla \varepsilon^{sp})_k\end{aligned} \tag{5.7}$$

Substitute (5.7) in (5.6), we have

$$\begin{aligned}\varepsilon_{k+1}^{sp} = \xi - 2&\left(\begin{bmatrix}\boldsymbol{\omega}_{p,k}^T & \boldsymbol{\omega}_{q,k}^T\end{bmatrix} - (\nabla\varepsilon^{sp})_k^T\begin{bmatrix}\eta_p\boldsymbol{I} & 0\\ 0 & \eta_q\boldsymbol{I}\end{bmatrix}\right)\begin{bmatrix}\Phi_{dp}\\ \Phi_{dq}\end{bmatrix}\\ &+\left(\begin{bmatrix}\boldsymbol{\omega}_{p,k}^T & \boldsymbol{\omega}_{q,k}^T\end{bmatrix} - (\nabla\varepsilon^{sp})_k^T\begin{bmatrix}\eta_p\boldsymbol{I} & 0\\ 0 & \eta_q\boldsymbol{I}\end{bmatrix}\right)\begin{bmatrix}\Phi_p & 0\\ 0 & \Phi_q\end{bmatrix}\\ &\cdot\left(\begin{bmatrix}\boldsymbol{\omega}_{p,k}\\ \boldsymbol{\omega}_{q,k}\end{bmatrix} - \begin{bmatrix}\eta_p\boldsymbol{I} & 0\\ 0 & \eta_q\boldsymbol{I}\end{bmatrix}(\nabla\varepsilon^{sp})_k\right)\end{aligned} \tag{5.8}$$

With some manipulation in mathematics and noting (5.4), we have

$$\begin{aligned}\varepsilon_{k+1}^{sp} = \varepsilon_k^{sp} &+ 2(\nabla\varepsilon^{sp})_k^T\begin{bmatrix}\eta_p\boldsymbol{I} & 0\\ 0 & \eta_q\boldsymbol{I}\end{bmatrix}\begin{bmatrix}\Phi_{dp}\\ \Phi_{dq}\end{bmatrix}\\ &-2\,(\nabla\varepsilon^{sp})_k^T\begin{bmatrix}\eta_p\boldsymbol{I} & 0\\ 0 & \eta_q\boldsymbol{I}\end{bmatrix}\begin{bmatrix}\Phi_p & 0\\ 0 & \Phi_q\end{bmatrix}\begin{bmatrix}\boldsymbol{\omega}_{p,k}\\ \boldsymbol{\omega}_{q,k}\end{bmatrix}\\ &+(\nabla\varepsilon^{sp})_k^T\begin{bmatrix}\eta_p\boldsymbol{I} & 0\\ 0 & \eta_q\boldsymbol{I}\end{bmatrix}\begin{bmatrix}\Phi_p & 0\\ 0 & \Phi_q\end{bmatrix}\begin{bmatrix}\eta_p\boldsymbol{I} & 0\\ 0 & \eta_q\boldsymbol{I}\end{bmatrix}(\nabla\varepsilon^{sp})_k\end{aligned} \tag{5.9}$$

Take the gradient operation upon (5.4) yields,

$$\begin{aligned}(\nabla\varepsilon^{sp})_k &= \begin{bmatrix}\nabla_p\\ \nabla_q\end{bmatrix}(\varepsilon^{sp})_k\\ &= -2\left(\begin{bmatrix}\Phi_{dp}\\ \Phi_{dq}\end{bmatrix} - \begin{bmatrix}\Phi_p & 0\\ 0 & \Phi_q\end{bmatrix}\begin{bmatrix}\boldsymbol{\omega}_{p,k}\\ \boldsymbol{\omega}_{q,k}\end{bmatrix}\right)\end{aligned} \tag{5.10}$$

Substitute (5.10) in (5.9), we obtain

$$\begin{aligned}\varepsilon_{k+1}^{sp} &= \varepsilon_k^{sp} + 2(\nabla\varepsilon^{sp})_k^T\begin{bmatrix}\eta_p\boldsymbol{I} & 0\\ 0 & \eta_q\boldsymbol{I}\end{bmatrix}\left(\begin{bmatrix}\Phi_{dp}\\ \Phi_{dq}\end{bmatrix} - \begin{bmatrix}\Phi_p & 0\\ 0 & \Phi_q\end{bmatrix}\begin{bmatrix}\boldsymbol{\omega}_{p,k}\\ \boldsymbol{\omega}_{q,k}\end{bmatrix}\right)\\ &\quad+(\nabla\varepsilon^{sp})_k^T\begin{bmatrix}\eta_p\boldsymbol{I} & 0\\ 0 & \eta_q\boldsymbol{I}\end{bmatrix}\begin{bmatrix}\Phi_p & 0\\ 0 & \Phi_q\end{bmatrix}\begin{bmatrix}\eta_p\boldsymbol{I} & 0\\ 0 & \eta_q\boldsymbol{I}\end{bmatrix}(\nabla\varepsilon^{sp})_k\\ &= \varepsilon_k^{sp} - (\nabla\varepsilon^{sp})_k^T\begin{bmatrix}\eta_p\boldsymbol{I} & 0\\ 0 & \eta_q\boldsymbol{I}\end{bmatrix}(\nabla\varepsilon^{sp})_k\\ &\quad+(\nabla\varepsilon^{sp})_k^T\begin{bmatrix}\eta_p\boldsymbol{I} & 0\\ 0 & \eta_q\boldsymbol{I}\end{bmatrix}\begin{bmatrix}\Phi_p & 0\\ 0 & \Phi_q\end{bmatrix}\begin{bmatrix}\eta_p\boldsymbol{I} & 0\\ 0 & \eta_q\boldsymbol{I}\end{bmatrix}(\nabla\varepsilon^{sp})_k\end{aligned} \tag{5.11}$$

After extending (5.11), finally we get

$$\begin{aligned}\varepsilon_{k+1}^{sp} &= \varepsilon_k^{sp} - \eta_p \left\| (\nabla_p \varepsilon^{sp})_k \right\|^2 - \eta_q \left\| (\nabla_q \varepsilon^{sp})_k \right\|^2 \\ &\quad + \eta_p^2 (\nabla_p \varepsilon^{sp})_k^T \Phi_p (\nabla_p \varepsilon^{sp})_k + \eta_q^2 (\nabla_q \varepsilon^{sp})_k^T \Phi_q (\nabla_q \varepsilon^{sp})_k\end{aligned} \tag{5.12}$$

Now, define

$$\begin{aligned}\boldsymbol{G}_p &= \left\| (\nabla_p \varepsilon^{sp})_k \right\|^2, \qquad \boldsymbol{H}_p = (\nabla_p \varepsilon^{sp})_k^T \Phi_p (\nabla_p \varepsilon^{sp})_k \\ \boldsymbol{G}_q &= \left\| (\nabla_q \varepsilon^{sp})_k \right\|^2, \qquad \boldsymbol{H}_q = (\nabla_q \varepsilon^{sp})_k^T \Phi_q (\nabla_q \varepsilon^{sp})_k\end{aligned} \tag{5.13}$$

Equation (5.12) becomes [Wan 1994a]

$$\varepsilon_{k+1}^{sp} = \varepsilon_k^{sp} - \eta_p \boldsymbol{G}_p - \eta_q \boldsymbol{G}_q + \eta_p^2 \boldsymbol{H}_p + \eta_q^2 \boldsymbol{H}_q \tag{5.14}$$

If the convergence factors are treated as time varying, that is taking $\eta_p(k)$ and $\eta_q(k)$ to replace the constant η_p and η_q, the error surface of the system is then a paraboloid which indicates that during each adaptation, the MSE is a quadratic function of $\eta_p(k)$ and $\eta_q(k)$. Therefore, a pair of optimal convergence factors, $\eta^*_p(k)$ and $\eta^*_q(k)$, exist at every iteration step which result in a unique global minimum of ε^{sp}_{k+1} for the quadratic performance surface. This relationship can be well illustrated by Fig.5.1.

We shall next derive the optimal convergence factors for the system. Assuming $\eta_p(k)$ and $\eta_q(k)$ are changing slowly during iteration, they can then be considered as independent to ε^{sp}_k and will only affect ε^{sp}_{k+1}. This assumption is reasonable in practice because slowly varying convergence factor will result in a smooth performance surface during adaptation. Following the assumption, $\eta_p(k)$ and $\eta_q(k)$ can be considered independent to the functions, $\boldsymbol{G}_p$, $\boldsymbol{G}_q$, $\boldsymbol{H}_p$ and $\boldsymbol{H}_q$, of ε^{sp}_k.

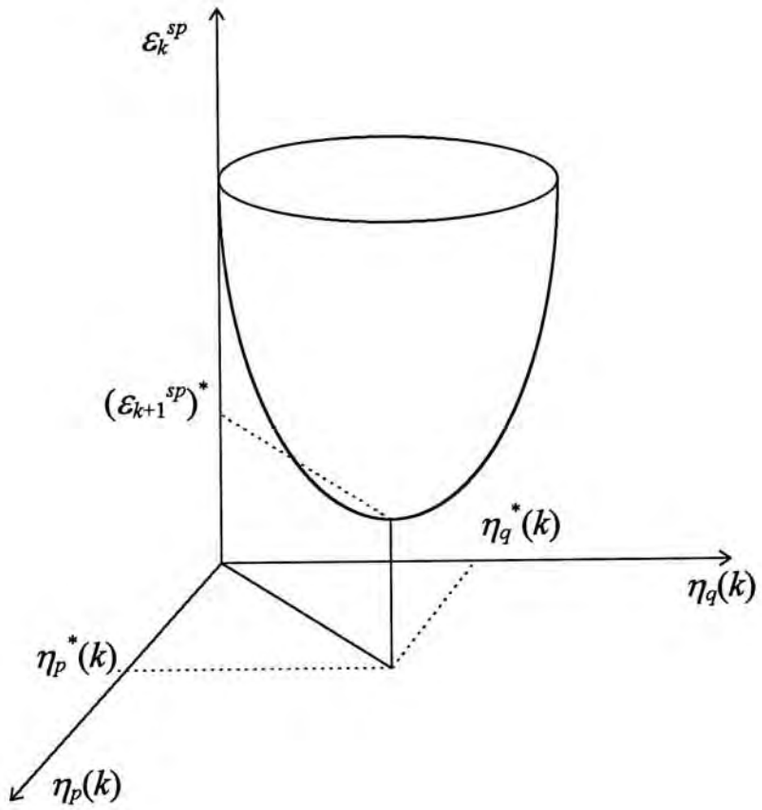


Fig. 5.1 Quadratic performance surface associated with optimal convergence factors, $\eta_p^*(k)$ and $\eta_q^*(k)$.

By partial differentiating ε^{sp}_{k+1} with respect to $\eta_p(k)$ and $\eta_q(k)$ and equating them to zero, we obtain

$$\frac{\partial \varepsilon^{sp}_{k+1}}{\partial \eta_p(k)} = \frac{\partial \varepsilon^{sp}_{k+1}}{\partial \eta_q(k)} = 0 \tag{5.15}$$

We can subsequently obtain the optimal value of the step sizes at instant k as follows,

$$\eta_p^*(k) = \frac{\boldsymbol{G}_p}{2\boldsymbol{H}_p} \qquad \text{and} \qquad \eta_q^*(k) = \frac{\boldsymbol{G}_q}{2\boldsymbol{H}_q} \tag{5.16}$$

In (5.16), the instantaneous optimum values, $\eta_p^*(k)$ and $\eta_q^*(k)$, depend only upon the second order statistical properties of the previous input signal associated with the individual branches. Referring to the definition in (5.13), it is noticed that at each iteration the convergence factor for individual adaptive branch varies in proportional to its respective norm of the gradient vector. Because gradient estimate predicts the changing rate of the MSE, the convergence factors will vary according to the estimate of the distance to the minimum MSE, thereby providing rapid convergence. $\boldsymbol{H}_p$ and $\boldsymbol{H}_q$ are the values of gradient estimates weighted by their associated autocorrelation matrices. That is, $\eta_p^*(k)$ and $\eta_q^*(k)$ are controlled by normalized gradient estimation. Substituting (5.16) into (5.14), we have

$$\begin{aligned}(\varepsilon^{sp}_{k+1})^* &= \varepsilon^{sp}_k - \frac{\boldsymbol{G}_p^2}{2\boldsymbol{H}_p} - \frac{\boldsymbol{G}_q^2}{2\boldsymbol{H}_q} + \frac{\boldsymbol{G}_p^2}{4\boldsymbol{H}_p^2}\boldsymbol{H}_p + \frac{\boldsymbol{G}_q^2}{4\boldsymbol{H}_q^2}\boldsymbol{H}_q \\ &= \varepsilon^{sp}_k - \frac{\boldsymbol{G}_p^2}{4\boldsymbol{H}_p} - \frac{\boldsymbol{G}_q^2}{4\boldsymbol{H}_q}\end{aligned} \tag{5.17}$$

Then we can express the "one step optimum MSE decrement" as

$$\begin{aligned}(\varepsilon^{sp}_{k+1} - \varepsilon^{sp}_k)^* &= -\frac{1}{4}\left(\frac{\boldsymbol{G}_p^2}{\boldsymbol{H}_p} + \frac{\boldsymbol{G}_q^2}{\boldsymbol{H}_q}\right) \\ &= -\delta_2 < 0\end{aligned} \tag{5.18}$$

The resultant in equation (5.18) is a negative value, which implies that with a proper choice of $\eta_p(k)$ and $\eta_q(k)$, the MSE will be descending as iteration goes on. Thus δ_2 can be regarded as a criterion governing the convergence behaviour, and the bigger the δ_2, the faster the convergence speed. Any choices other then $\eta_p^*(k)$ and $\eta_q^*(k)$ will get a smaller value of δ_2 which results in a slower convergence speed at instant k [Wan 1995b].

Now, let us consider what will happen when identical convergence factor, $\eta(k) = \eta_p(k) = \eta_q(k)$, is assigned to both adaptive branches. In this case, (5.14) becomes

$$\varepsilon_{k+1}^{sp} = \varepsilon_k^{sp} - \eta(k)\boldsymbol{G}_p - \eta(k)\boldsymbol{G}_q + \eta^2(k)\boldsymbol{H}_p + \eta^2(k)\boldsymbol{H}_q \tag{5.19}$$

Taking derivative of (5.19) with respective to $\eta(k)$,

$$\frac{\partial \varepsilon_{k+1}^{sp}}{\partial \eta(k)} = -\boldsymbol{G}_p - \boldsymbol{G}_q + 2\eta(k)(\boldsymbol{H}_p + \boldsymbol{H}_q) \tag{5.20}$$

This yields a single optimal convergence factor which is given by

$$\eta^*(k) = \frac{\boldsymbol{G}_p + \boldsymbol{G}_q}{2(\boldsymbol{H}_p + \boldsymbol{H}_q)} \tag{5.21}$$

In this case, the one step decrement of the MSE becomes

$$(\varepsilon_{k+1}^{sp} - \varepsilon_k^{sp})^* = -\frac{1}{4}\frac{(\boldsymbol{G}_p + \boldsymbol{G}_q)^2}{\boldsymbol{H}_p + \boldsymbol{H}_q} = -\delta_1 \tag{5.22}$$

where δ_1 is also a criterion governing the convergence behaviour, which reflects the convergence speed of the adaptive system. Compare the values of δ_2 and δ_1 by subtracting (5.22) from (5.18), it gives

$$\delta_2 - \delta_1 = \frac{\boldsymbol{G}_p^2\boldsymbol{H}_q^2 + \boldsymbol{G}_q^2\boldsymbol{H}_p^2 - 2\boldsymbol{H}_p\boldsymbol{H}_q\boldsymbol{G}_p\boldsymbol{G}_q}{4\boldsymbol{H}_p\boldsymbol{H}_q(\boldsymbol{H}_p + \boldsymbol{H}_q)} \tag{5.23}$$

Since G_p, G_q, H_p and H_q are all positive definite, therefore the numerator

$$G_p^2 H_q^2 + G_q^2 H_p^2 - 2H_p H_q G_p G_q = (G_p H_q - G_q H_p)^2 \geq 0 \tag{5.24}$$

that is

$$\delta_2 - \delta_1 \geq 0 \tag{5.25}$$

That means, by using individual optimal convergence factors, $\eta_p^*(k)$ and $\eta_q^*(k)$, for associated branches, TRSPAF always achieves a faster convergence speed than that of using a single convergence factor $\eta^*(k)$ [Wan 1995b].

When δ_2 -δ_1=0, (5.24) equals zero, which implies that

$$\frac{G_p}{H_p} = \frac{G_q}{H_q} \tag{5.26}$$

Obviously, the signals feed to both branches now possess exactly the same second order statistical properties and the input signal is in fact with eigenvalue spread equals to unity. Consequently, the split technique will provide no improvement to the convergence dynamics of the system [Wan 1995b].

5.3 Tracking Optimal Convergence Factors for TRSPAF

The optimal convergence factors $\eta_p^*(k)$ and $\eta_q^*(k)$ derived in the previous section are functions of the second order statistical variables. During adaptation, since G_p, G_q, H_p or H_q is varying with the input signal, especially when the statistical characteristics of the data are real time changing, their exact values are not available. It is required to track their variations in order to calculate the associated optimal

convergence factors and to warrant a good adaptation performance while at the same time keeping simplicity of the computation.

5.3.1 Tracking Optimal Convergence Factors for Gradient-Based Algorithms

Recall the gradient operation of branch p, with respect to $\omega_{p,k}$, in transform domain

$$(\nabla_p \varepsilon^{sp})_k = \frac{\partial E[e^2(k)]}{\partial \omega_{p,k}} \tag{5.27}$$

By changing the order of differentiation with expectation, we have

$$(\nabla_p \varepsilon^{sp})_k = E[-2e(k)\boldsymbol{f}_{p,k}] \tag{5.28}$$

Substitute (5.28) in (5.16)

$$\eta_p^*(k) = \frac{\boldsymbol{G}_p}{2\boldsymbol{H}_p} = \frac{E[-2e(k)\boldsymbol{f}_{p,k}]^T E[-2e(k)\boldsymbol{f}_{p,k}]}{2E[-2e(k)\boldsymbol{f}_{p,k}]^T \Phi_p E[-2e(k)\boldsymbol{f}_{p,k}]} \tag{5.29}$$

When the convergence factor is sufficiently small, $e(k)$ approaches to its steady state value very slowly, thus it can be assumed independent with $\boldsymbol{f}_{p,k}$. That is,

$$E[e(k)\boldsymbol{f}_{p,k}] = E[e(k)]E[\boldsymbol{f}_{p,k}] \tag{5.30}$$

Then (5.29) reduces to

$$\eta_p^*(k) = \frac{E[\boldsymbol{f}_{p,k}]^T E[\boldsymbol{f}_{p,k}]}{2E[\boldsymbol{f}_{p,k}]^T \Phi_p E[\boldsymbol{f}_{p,k}]} \tag{5.31}$$

It has been proved in Chapter 4 that for an orthogonal transform, when the decaying time of the input autocorrelation is sufficiently small compared to the filter length, the

autocorrelation matrix is approximately diagonal. Accordingly, the off-diagonal elements of Φ_p can be neglected. Therefore, the estimation of $\eta_p^*(k)$ can be evaluated as

$$\eta_p^*(k) = \frac{E[\boldsymbol{f}_{p,k}]^T E[\boldsymbol{f}_{p,k}]}{2E[\boldsymbol{f}_{p,k}]^T E[diag\{f_{p,1}^2(k), f_{p,2}^2(k), \cdots, f_{p,N}^2(k)\}]E[\boldsymbol{f}_{p,k}]} \tag{5.32}$$

where *diag*{.} denotes a diagonal matrix. For a tapped delay line input

$$E[f_{p,i}(k)] = E[f_{p,j}(k)] \qquad \text{for all} \quad i, j \tag{5.33}$$

(5.32) simplifies to

$$\eta_p^*(k) = \frac{1}{2\alpha_p^*(k)} \tag{5.34}$$

where

$$\alpha_p^*(k) = E\left[\sum_{i=1}^{N} f_{p,i}^2(k)\right] = E\left[\left\|\boldsymbol{f}_{p,k}\right\|^2\right] \tag{5.35}$$

Similarly, we can derive the tracking optimal convergence factor for the branch $\boldsymbol{q}$,

$$\eta_q^*(k) = \frac{1}{2\alpha_q^*(k)} \tag{5.36}$$

where

$$\alpha_q^*(k) = E\left[\sum_{i=1}^{N} f_{q,i}^2(k)\right] = E\left[\left\|\boldsymbol{f}_{q,k}\right\|^2\right] \tag{5.37}$$

5.3.2 Tracking Optimal Convergence Factors for LMS Algorithm

Now, we introduce the optimal convergence factor tracking method for the LMS algorithm and change the parameter updating equation as follows,

$$\omega_{p,k+1} = \omega_{p,k} + \frac{1}{\alpha_p^*(k)} f_{p,k} e(k) \tag{5.38}$$

It is noticed from (5.38) that

$$\eta_p^*(k) = \frac{1}{2\alpha_p^*(k)} = \frac{1}{2tr(\Phi_p)} < \frac{1}{tr(\Phi_p)} \tag{5.39}$$

which implies the stability condition for conventional LMS algorithm still holds. To avoid a large misadjustment, a damping factor, $0 < \beta < 1$, is introduced in the adaptation and (5.38) is modified as follows [Wan 1995b],

$$\omega_{p,k+1} = \omega_{p,k} + \frac{\beta}{\alpha_p^*(k)} f_{p,k} e(k) \tag{5.40}$$

Simulation results have shown that the selection of β is not very stringent. Similarly, the updating equation for $\omega_{q,k}$ can also be written as

$$\omega_{q,k+1} = \omega_{q,k} + \frac{\beta}{\alpha_q^*(k)} f_{q,k} e(k) \tag{5.41}$$

The values of $\alpha_p^*(k)$ and $\alpha_q^*(k)$ are actually the statistical expectation of the norm of the previous input signal, they can be estimated by computing from

$$\alpha_p^*(k+1) = \rho\alpha_p^*(k) + (1-\rho)\left\| f_{p,k} \right\|^2 \tag{5.42}$$

and

$$\alpha_q^*(k+1) = \rho\alpha_q^*(k) + (1-\rho)\left\| f_{q,k} \right\|^2 \tag{5.43}$$

where $0 \le \rho \le 1$ is a smoothing constant which controls estimation accuracy and tracking capability of time variations.

5.4 Comparison of Optimal Convergence Factor Tracking Method with Self-Orthogonalizing Method

In this section, we compare the performance of the optimal convergence factor tracking method with the self-orthogonalizing method, when applied to LMS adaptation. In Chapter 4, we have derived the self-orthogonalizing LMS algorithm for TRSPAF, its parameter updating equation for branch $\boldsymbol{p}$ is in the form of

$$\boldsymbol{\omega}_{p,k+1} = \boldsymbol{\omega}_{p,k} + 2\eta_p \Lambda_p^{-2}\, e_p(k) \boldsymbol{f}_{p,k} \tag{5.44}$$

The convergence behaviour can be investigated by the equation of parameter estimation error

$$E[\Delta\boldsymbol{\omega}_{p,k+1}] = (\boldsymbol{I} - 2\eta_p \Lambda_p^{-2}\, \Phi_p)\, E[\Delta\boldsymbol{\omega}_{p,k}] \tag{5.45}$$

For optimal convergence factor tracking method, we can also write out its equation of parameter estimation error as follows,

$$E[\Delta\boldsymbol{\omega}_{p,k+1}] = (\boldsymbol{I} - \frac{\beta}{\alpha_p^*(k)}\, \Phi_p)\, E[\Delta\boldsymbol{\omega}_{p,k}] \tag{5.46}$$

In (5.45), estimate of the inverse matrix Λ_p^{-2} is employed to identify the autocorrelation matrix Φ_p. While in (5.46), the estimate of norm α_p^* is used to normalize Φ_p. Both methods are derived dependent on the assumption that the residual elements in the autocorrelation matrix, after orthogonal transform, are fairly small that it can be ignored. Actually, any transformation except KLT cannot exactly whiten the spectrum of the signal, therefore, both methods are only *suboptimal*. This is the reason we call the later o*ptimal tracking* method. In (5.45), the elements in Λ_p^2 are rather sensitive to input data for every parameter. If a power estimate falls below

an acceptable level, adaptation of the corresponding parameter is disabled in order to ensure stability. However, α_p^* in (5.46) is an ensemble of the power estimate. It is then less sensitive to the larger fluctuation of *individual* signal. For example, if σ_i^2 falls to a very small value, it will cause the adaptation of the ith parameter in (5.45) to cease, but (5.46) still works regularly in most cases. Hence, optimal convergence factor tracking method has a wider stability range and a robust performance behaviour. Therefore, it can be seen from the formulae that for transformation that can well whiten the signal, or equivalently Φ_p is rather precisely diagonalized, self-orthogonalizing method will perform better. But for more general cases when the signal is not ideally whitened or not exactly known, optimal convergence factor tracking method is preferred.

If the identical steady state MSE is confined to both methods, we will have, from (5.45) and (5.46),

$$2\eta_p \Lambda_p^{-2} = \frac{\beta}{\alpha_p^*(k)} \boldsymbol{I} \tag{5.47}$$

For the TDL system, $\sigma_i^2 = \sigma_j^2$ $(i \neq j)$, (5.47) reduces to

$$\frac{2\eta_p}{\sigma_i^2} = \frac{\beta}{N\sigma_i^2} \tag{5.48}$$

That is

$$\beta = 2N\eta_p \tag{5.49}$$

In practice, N is a large integer, which implies that the adaptive system is less sensitive to changes of β than to changes of η_p. Such behaviour also holds in branch $\boldsymbol{q}$. This distinct property will be demonstrated by computer simulations.

5.5 Computer Simulation Results

Extensive simulations have been done with different transformations to verify the analytical results and to evaluate the convergence behaviour of the transform domain split-path adaptive filter. An application of a DCT-domain adaptive line enhancer is presented here as a typical example [Wan 1994a].

The block diagram of the adaptive line enhancer is shown in Fig. 5.2. Here, the input signal is given by

$$\begin{aligned} x(k) &= d(k) + n(k) \\ &= 0.1\cos(\frac{\pi}{15}kT) + \cos(\frac{5\pi}{16}kT) + n(k) \end{aligned} \tag{5.50}$$

where the desired sine wave $d(k)$ is immersed in a white Gaussian noise, $n(k)$, of variance of 0.1. The sampling time is T=0.01 second. A time delay Δ is applied to the contaminated input signal from which a reference signal is obtained and it has the effect of decorrelating the broadband noise added to $d(k)$ and $d(k-\Delta)$.

In this example, the filter length M is set to 16 and the delay unit Δ has 7 delay taps. DCT is used to generate the transform domain input signal and LMS algorithm is employed with three different kinds of adaptive method, namely,

(1) non-split-path self-orthogonalizing method,

(2) split-path self-orthogonalizing method, and

(3) split-path optimal convergence factor tracking method.

The damping factor for optimal convergence factor tracking method was set as $\beta = 0.1$. In experiments, all data were obtained from an average of 500 independent simulation runs. Fig. 5.3 shows the learning trajectories and Fig. 5.4 shows the output signals of the three trials with the same assigned steady state MSE.

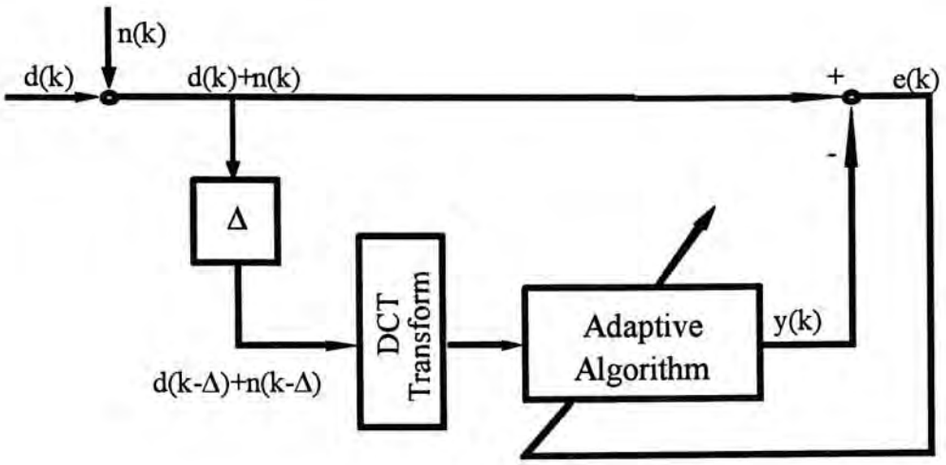


Fig. 5.2 An adaptive line enhancer.

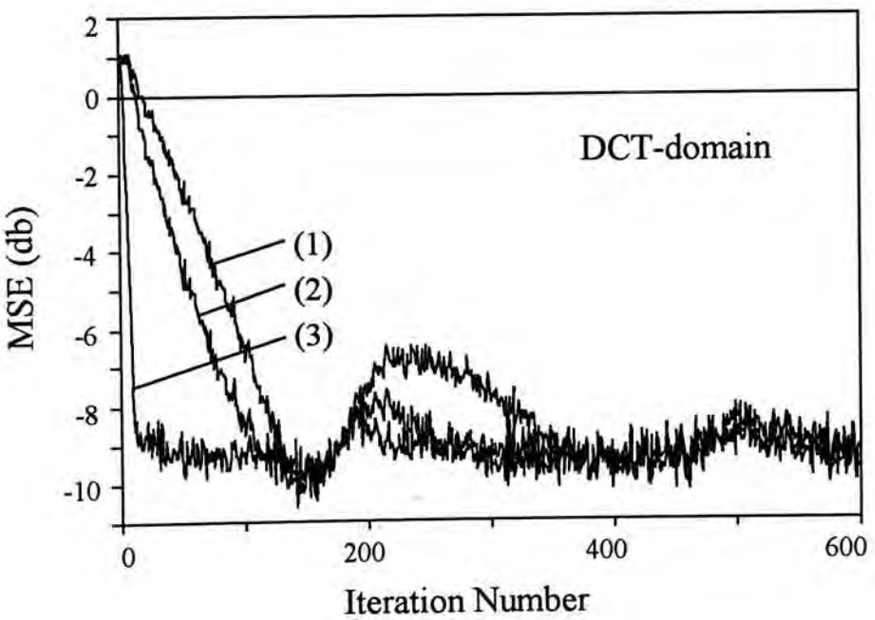


Fig. 5.3 Comparison of learning trajectories in DCT-domain:

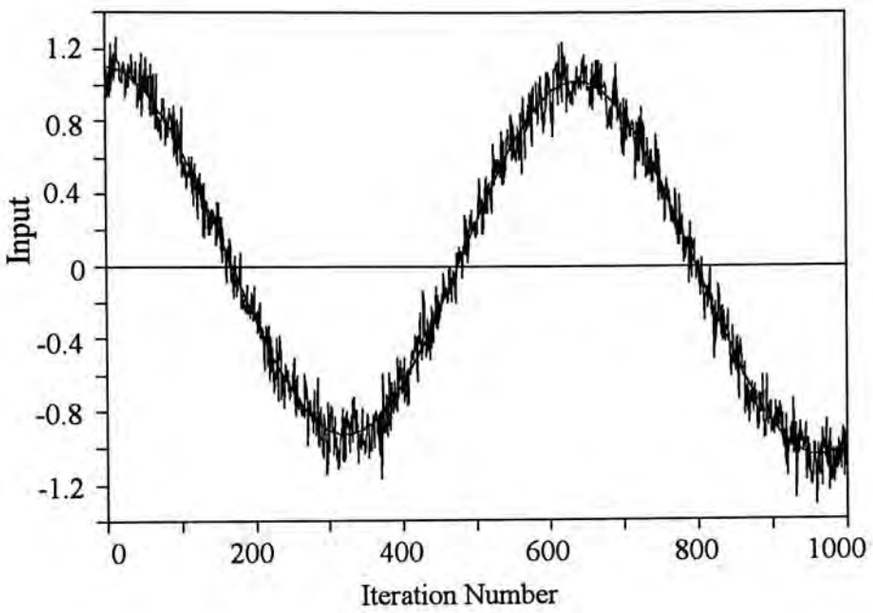
(1). Nonsplit-path, η =0.003; (2). Split-path, $\eta_p = \eta_q$ =0.003, ρ =0.96;

(3). Split-path, with optimal convergence factors, ρ =0.96, β =0.1.

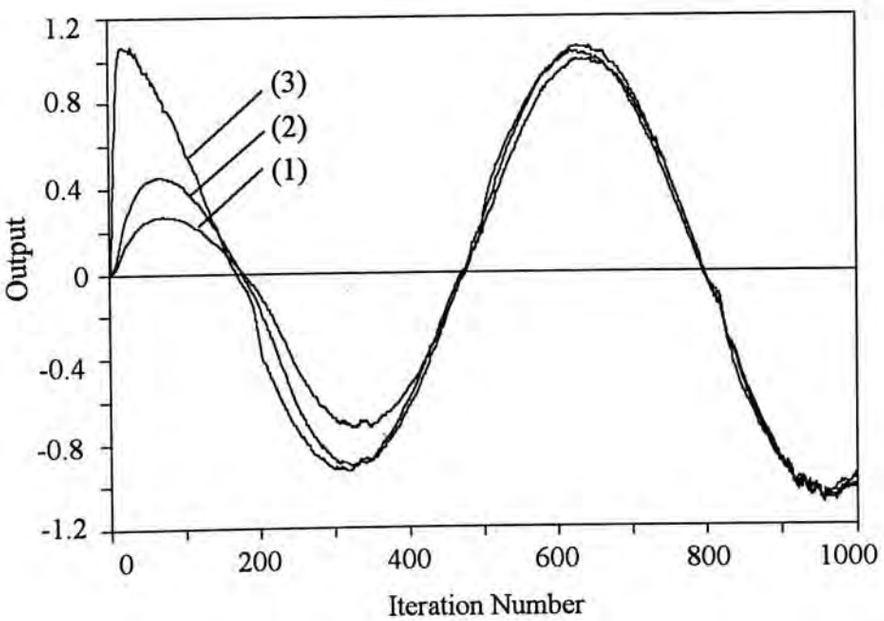
For self-orthogonalizing methods (1) and (2), an identical convergence factor $\eta=\eta_q=\eta_p=0.003$ is used, it can be seen that the split structure converges at a much faster rate. Comparing the split-path filter with adaptive methods (2) and (3), we note that the MSE obtained by using the optimum convergence factor tracking method had dropped sharply to a value of -9 dB at less than 50 iterations with only 1 dB ripple transient in the first period of the sine wave. This confirms that TRSPAF with adaptive tracking of the optimum convergence factor can produce a much better convergence dynamics.

Fig. 5.5 shows the effect of choosing different values of damping factor β on the convergence behaviour for method (3). Assigning β to 0.05, 0.08, 0.1 and 0.3 respectively, their corresponding steady state MSE are almost the same and only very little differences are noted within the first 30 iterations. This illustrates that the selection of β is not very strict. In addition, it appears that a bigger β will result in faster convergence speed, but small fluctuation will be caused as well. To avoid such fluctuation, as a rule of thumb, it is usually chosen β within the range of

$$0.01 < \beta < 0.5 \tag{5.51}$$



(a)



(b)

Fig. 5.4 Comparison of outputs from different methods

(a). Desired signal immersed in a white noise; (b). Outputs from different methods.

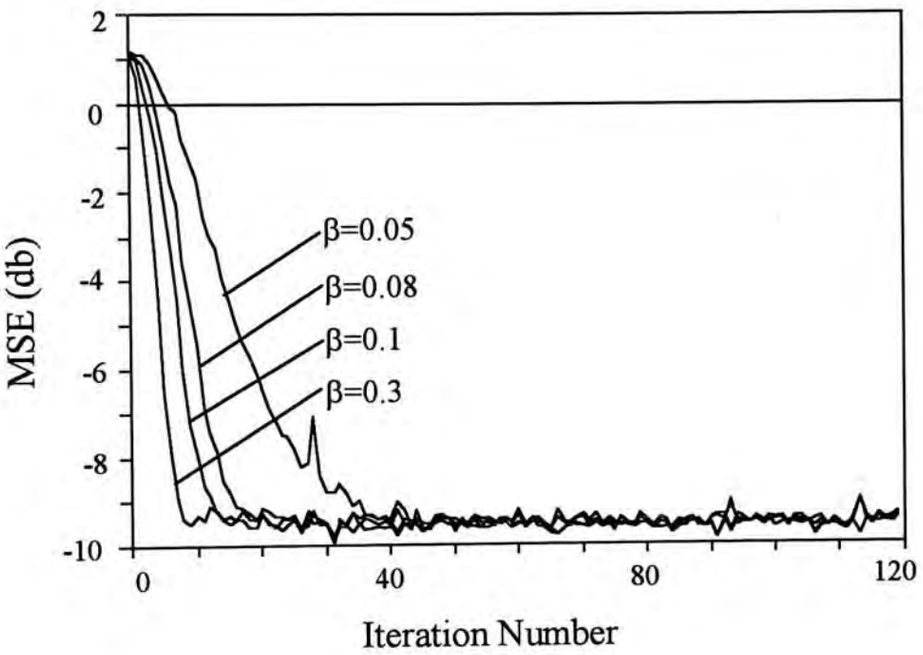


Fig. 5.5 Effect of different damping factor β

5.6 Summary

In this chapter, we have derived a pair of optimal convergence factors for transform domain adaptive algorithm which is configured in a split-path filter structure. During the adaptation, the convergence factor keeps varying in proportional to the norm of the respective gradient vector which is actually the prediction of the distance to the minimum MSE thereby rapid convergence is provided. The proposed algorithm is very simple to implement. A practical optimal convergence factor tracking method is also proposed to facilitate the gradient-based algorithm, such as the LMS algorithm, to operate in a changing environment. By comparison with other methods, the optimal convergence factor tracking method has shown to be less sensitive to the misadjustment and have a wider stability range and a robust performance behaviour than the self-orthogonalizing method.

Chapter 6 A UNIFICATION BETWEEN SPLIT-PATH ADAPTIVE FILTERING AND DISCRETE WALSH TRANSFORM ADAPTATION

In Chapter 3, we have developed a multi-stage split-path adaptive algorithm and have shown by computer simulations that it has inherent characteristics that are similar to adaptation in the Walsh transform domain. In this Chapter, a new set of Walsh function will be defined in terms of the multi-stage splitting matrix (SM). We shall illustrate that this is essentially a complete set of Walsh function and a unified perspective of the proposed split-path adaptive filtering method with discrete Walsh transform adaptation is thus established. The intrinsic properties of the newly defined Walsh function are analyzed and its relationship with other well-defined Walsh functions and the associated conversion rules will be discussed. We shall also demonstrate that the realization of fast DWT by SM-ordered Walsh function is much more efficient than other fast DWT algorithms [Wan 1995a] [Wan 1995c].

6.1 Introduction

We have shown in Chapter 3 that any $M=2^L$-coefficient FIR filter can be decomposed to M parallel subunits by a multi-stage splitting operation. While the

recomposed parallel input vector, $\boldsymbol{g}_k$, to the adaptive subunits can be obtained from the original input vector by a L-step matrix multiplication

$$\boldsymbol{g}_k = \boldsymbol{U}_L \boldsymbol{U}_{L-1} \cdots \boldsymbol{U}_1 \boldsymbol{x}_k \tag{6.1}$$

For a typical 8 coefficients system, the new input vector is given by

$$\boldsymbol{g}_k = \boldsymbol{U}_3 \boldsymbol{U}_2 \boldsymbol{U}_1 \boldsymbol{x}_k = \begin{bmatrix} 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \boldsymbol{x}_k = \boldsymbol{H}_8 \boldsymbol{x}_k \tag{6.2}$$

Let us compare $\boldsymbol{H}_8$ with an 8×8 Walsh-Hadamard matrix expression [Beauchamp 1984],

$$\boldsymbol{W}_h = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & -1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{bmatrix} \tag{6.3}$$

it can be found that the elements in both matrices are exactly the same except there are some permutation changes in rows. This distinct feature gives an inspiration that a unique relationship might exist between the multi-stage splitting matrix and DWT.

6.2 A New Ordering of the Walsh Functions

The Walsh functions form an ordered set of rectangular waveforms taking only two amplitude values +1 and -1 and are orthogonal sets of function [Beauchamp 1984]. The Walsh function is usually expressed as

$$\text{Wal}(k,t), \qquad k = 0,1,\cdots,M-1 \tag{6.4}$$

where k is an ordering number related to frequency, and t is the sampling time. The Walsh function series can be derived in many different ways, such as from recursive relations [Chien 1975], from products of Rademacher functions [Lackey 1971], through the Hadamard matrix [Rushforth 1969] and by the use of Boolean synthesis [Gibbs 1970]. All these derivations are, of course, mathematical processes for which computational algorithms can be developed and the series produced by using he digital computer or obtained directly by using digital logic. However, owing to the simplicity of the fast Walsh transform algorithm and the speed with which this can be implemented on the digital machine, each of the definition has its own particular advantages [Beauchamp 1984]. In this section, we shall define a new set of Walsh functions, by means of the splitting matrix, which is called the splitting matrix (SM) Walsh function expression.

Definition: A set of $M=2^L$ SM-ordered Walsh functions, $\text{Wal}_{sm}(k, t)$, can be defined as the rows of a transform matrix $\boldsymbol{H}_{sm}$, which is the successive product of $M \times M$ splitting matrices $\boldsymbol{U}_i$ ($i=1,...,L$),

$$\boldsymbol{H}_{sm} \triangleq \prod_{i=1}^{L} \boldsymbol{U}_i = \boldsymbol{U}_L \, \boldsymbol{U}_{L-1} \cdots \boldsymbol{U}_2 \boldsymbol{U}_1 \tag{6.5}$$

where $\boldsymbol{U}_i$ is defined by,

$$\boldsymbol{U}_i \triangleq \begin{bmatrix} \boldsymbol{u}_i & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \boldsymbol{u}_i \end{bmatrix} \qquad i = 1,\ 2,\ \cdots,\ L \tag{6.6}$$

The diagonal block element in $\boldsymbol{U}_i$ is of the form

$$\boldsymbol{u}_i \triangleq \begin{bmatrix} \boldsymbol{I}_i & -\boldsymbol{J}_i \\ \boldsymbol{I}_i & \boldsymbol{J}_i \end{bmatrix} \qquad i = 1,\ 2,\ \cdots,\ L \tag{6.7}$$

where $\boldsymbol{I}_i$ and $\boldsymbol{J}_i$ are $(M/2^i)\times(M/2^i)$ identity matrix and anti-diagonal identity matrix. The arguments, k and t, are also referred as the row number and column number of $\boldsymbol{H}_{sm}$, respectively. Let $\mathrm{Wal}_h(k,t)$ denotes the Walsh function defined through Hadamard matrix, it is easy to check that inter-relationships exist among $\mathrm{Wal}_{sm}(k,t)$, $\mathrm{Wal}_h(k,t)$ and $\mathrm{Wal}(k,t)$. This conversion is listed in Table 6.1 (for $M = 8$).

SM Walsh	Hadamard Walsh	Walsh
$\mathrm{Wal}_{sm}(0,t)$	$\mathrm{Wal}_h(1,t)$	$\mathrm{Wal}(7,t)$
$\mathrm{Wal}_{sm}(1,t)$	$\mathrm{Wal}_h(2,t)$	$\mathrm{Wal}(3,t)$
$\mathrm{Wal}_{sm}(2,t)$	$\mathrm{Wal}_h(7,t)$	$\mathrm{Wal}(5,t)$
$\mathrm{Wal}_{sm}(3,t)$	$\mathrm{Wal}_h(4,t)$	$\mathrm{Wal}(1,t)$
$\mathrm{Wal}_{sm}(4,t)$	$\mathrm{Wal}_h(5,t)$	$\mathrm{Wal}(6,t)$
$\mathrm{Wal}_{sm}(5,t)$	$\mathrm{Wal}_h(6,t)$	$\mathrm{Wal}(2,t)$
$\mathrm{Wal}_{sm}(6,t)$	$\mathrm{Wal}_h(3,t)$	$\mathrm{Wal}(4,t)$
$\mathrm{Wal}_{sm}(7,t)$	$\mathrm{Wal}_h(0,t)$	$\mathrm{Wal}(0,t)$

Table. 6.1. Conversion for different Walsh functions.

Apparently, the rows of $\boldsymbol{H}_{sm}$ have constructed a complete set of Walsh function. Since

$$\boldsymbol{U}_i\ \boldsymbol{U}_i^T = 2\boldsymbol{I} \qquad i = 1,2,\ \cdots,\ L \tag{6.8}$$

we have

$$\boldsymbol{H}_{sm} \cdot \boldsymbol{H}_{sm}^{T} = (\prod_{i=L}^{1} \boldsymbol{U}_i)(\prod_{i=L}^{1} \boldsymbol{U}_i)^{T} = 2^{L} \boldsymbol{I} = M\boldsymbol{I} \tag{6.9}$$

This means that $\boldsymbol{H}_{sm}/\sqrt{M}$ is a " normalized orthogonal matrix". Hence, (6.1) actually represents a Walsh transform of the input sequence which implies that time domain multi-stage split-path adaptive filtering is equivalent to DWT domain adaptation. Hence we conclude that the continuous split-path structure has a unification with discrete Walsh transformation [Wan 1995c] [Ching 1995].

6.3 Relationship Between SM-Ordered Walsh Function and Other Walsh Functions

The difference between all kinds of Walsh functions is generally reflected by their ordering number k. The function $\mathrm{Wal}_{sm}(k,t)$ or $\boldsymbol{H}_{sm}$ can be uniquely mapped to other Walsh functions. The conversion from one function to another can be easily facilitated by using *Gray code* and/or *Bit reversal regulation* [Beauchamp 1984] or/and complementary operation. This conversion is shown in Fig. 6.1 by the solid line arrows. In addition, if we exchange the symmetric and antisymmetric parts in the definition of the splitting matrix $\boldsymbol{U}_i$, that is to redefine $\boldsymbol{u}_i$ as

$$\boldsymbol{u}_i = \begin{bmatrix} \boldsymbol{I}_i & \boldsymbol{J}_i \\ \boldsymbol{I}_i & -\boldsymbol{J}_i \end{bmatrix} \qquad i = 1,\ 2,\ \cdots,\ L \tag{6.10}$$

the SM ordering can be linked up with Hadamard ordering by Gray code as well, which is shown in Fig. 6.1 by the dotted line arrow.

Fig. 6.2 uses an example to illustrate the conversion of a Walsh ordering k_w to a SM ordering k_{sm}, when $M = 8$.

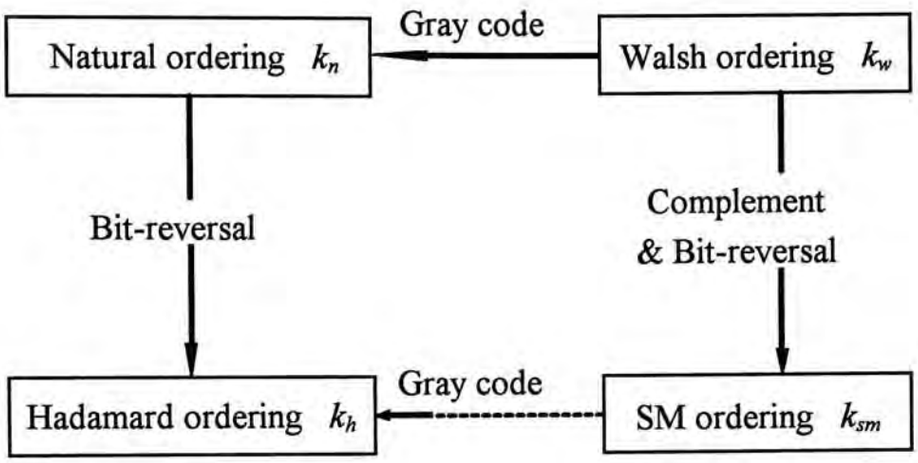


Fig. 6.1 Relationship between different Walsh function definitions.

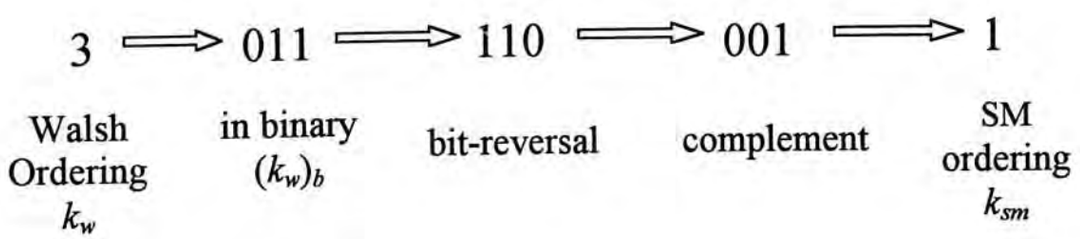


Fig. 6.2 Converting from Wal(k,t) to Wal$_{sm}$(k,t), (M=8, k=3).

Table. 6.2 gives a complete progressive procedure for an 8-order converting relationship between Walsh ordinal number k_w, natural ordinal number k_n, Hadamard ordinal number k_h and SM-ordinal number k_{sm}. In this table, subscript b denotes binary expression.

k_h	k_n	k_w	$(k_w)_b$	bit-reversal to $(k_w)_b$	complement + bit-reversal to $(k_w)_b$	k_{sm}
0	0	0	000	000	111	7
4	1	1	001	100	011	3
6	3	2	010	010	101	5
2	2	3	011	110	001	1
3	6	4	100	001	110	6
7	7	5	101	101	010	2
5	5	6	110	011	100	4
1	4	7	111	111	000	0

Table. 6.2 Example for converting between k_h, k_n, k_w and k_{sm} (M=8).

Due to the bit-reversal operation, the order for the corresponding sequency functions in the Table. 6.2 will be different for different value of M. Therefore, additional subscript M is added. That is k_{sm} can be rewritten as $k_{sm,M}$, such as $k_{sm,2}$, $k_{sm,4}$, $k_{sm,8}$, etc. Table 6.3 lists some examples of $k_{sm,M}$ associated with k_w when M equals 4, 8 and 16.

k_w	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$k_{sm,4}$	3	1	2	0												
$k_{sm,8}$	7	3	5	1	6	2	4	0								
$k_{sm,16}$	15	7	11	3	13	5	9	1	14	6	10	2	12	4	8	0

Table 6.3. Conversion between $k_{sm,4}$, $k_{sm,8}$, $k_{sm,16}$ and k_w.

The SM Walsh function matrix $\boldsymbol{H}_{sm}$ possesses most of the important properties as for other Walsh transform matrices. The $\boldsymbol{H}_{sm}$ can be divided into two parts, the upper rows are antisymmetric about its mid-point while the lower rows are symmetric about its mid-point. The characteristics of the splitting matrix enables the realization of fast discrete Walsh transform (DWT) to be more intuitive and simple to implement. Next, we shall compare the fast DWT algorithms by using Walsh-Hadamard matrix and SM Walsh function matrix. The DWT of the time domain sequence $\boldsymbol{x}_k$ may be expressed as

$$\boldsymbol{g}_k = \boldsymbol{W}_M \boldsymbol{x}_k \tag{6.11}$$

where $\boldsymbol{W}_M$ is the matrix representation of DWT, $\boldsymbol{g}_k$ is a transformed output data string. The transformation is carried out by multiplying the input data string $\boldsymbol{x}_k$ with the elements of $\boldsymbol{W}_M$ to obtain $\boldsymbol{g}_k$. Since the products represent multiplication by +1 or -1, this process therefore requires $M(M\text{-}1)$ additions or subtractions. However, by using some manipulations on the matrix, the task can be simplified. Among the different kinds of fast DWT algorithms [Beauchamp 1984], the Walsh-Hadamard method is the

easiest and has been widely used. The flow chart for computing an 8×8 fast Walsh-Hadamard transform is shown in Fig. 6.3.

While for a discrete SM Walsh transform, the computation flow chart, Fig. 6.4, can be drawn directly from its definition. In Fig. 6.4, $\boldsymbol{U}_1$, $\boldsymbol{U}_2$ and $\boldsymbol{U}_3$ represent the first, second and third independent computing steps respectively.

Intriguingly, both flow charts have the butterfly structure that is similar to an FFT algorithm. The operations are divided into $L=\log_2 M$ steps and each step includes 2^{i-1} calculative "groups". In addition, it is easy to find that the total amount of additions or subtractions are exactly the same for these algorithms. In deriving the fast Walsh-Hadamard algorithm, some matrix simplifying operations are required, and the properties of Hadamard matrix is not explicitly reflected in the flow chart. However, the definition of SM Walsh function is itself already a fast algorithm. Because there are no redundant information but only two irreducible elements (1 or -1) being located in every row of $\boldsymbol{U}_i$, each row represents one addition or subtraction operation. Therefore, matrix $\boldsymbol{U}_i$ has described the ith computing step of the fast DWT. The fast DWT flow chart, Fig. 6.4, can be drawn directly according to the splitting matrix $\boldsymbol{U}_i$ without any additional analysis or simplifying procedures. Therefore, the SM-ordered Walsh function provides a straightforward implementation of a fast discrete Walsh transformation [Wan 1995c].

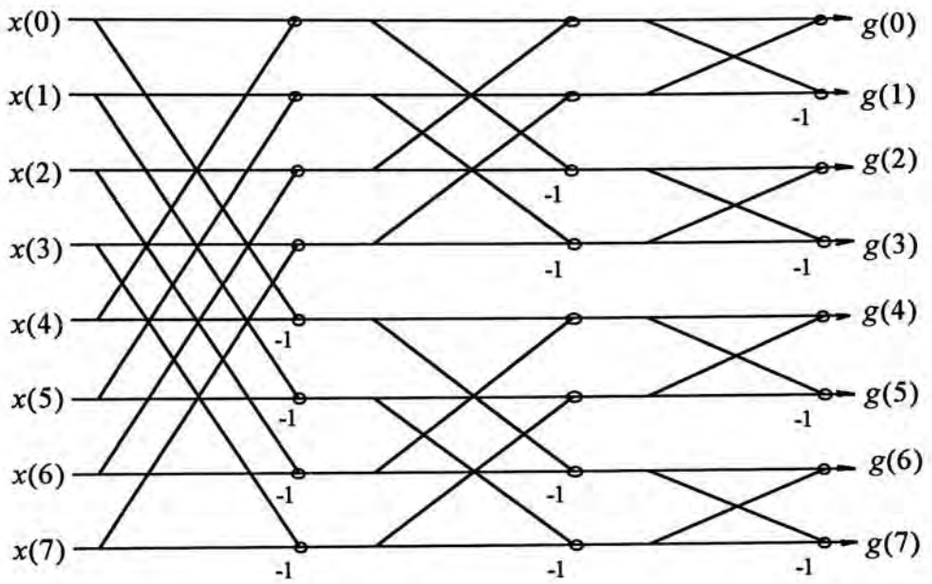


Fig. 6.3. Fast Walsh-Hadamard transform flow chart (M=8).

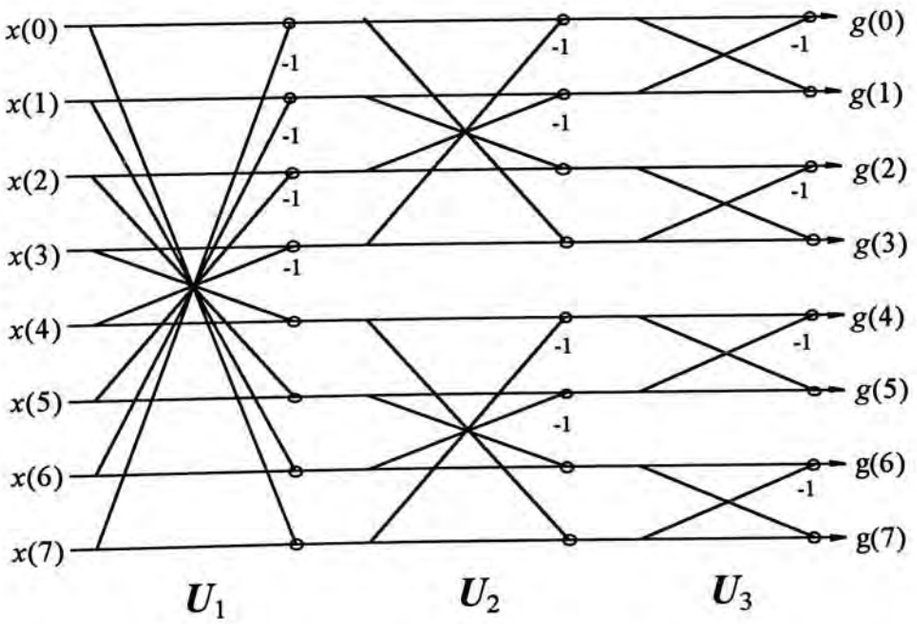


Fig. 6.4. Fast SM Walsh transform flow chart (M=8).

6.4 Computer Simulation Results

The performance of the multi-stage split-path adaptive filtering and its unification with DWT adaptation have already been described in Chapter 3. In this section, a computer simulation in the context of AR modeling is presented to further illustrate the merits of the proposed method. An 8-order AR process example being used is obtained from [Ching 1995]

$$\begin{aligned} x(k) = {} & 0.09x(k-1)+0.12x(k-2)+0.6x(k-3)+0.01x(k-4) \\ & +0.51x(k-5)+0.46x(k-6)+0.22x(k-7)+0.31x(k-8)+n(k) \end{aligned} \tag{6.12}$$

where $n(k)$ is a Gaussian random process with variance equals to 0.1 and M equals to 8. We have applied different levels of splitting to the adaptive filter and their convergence dynamics are compared with the conventional TDL approach. Fig. 6.5 compares the learning curves of these algorithms and all the data were obtained from an average of 10000 independent runs. To reach a value of 0.055 MSE, about 400, 300 and 250 iterations are needed for conventional LMS, 1-level split LMS and 2-level LMS respectively. While only less than 200 iterations has been found to be required by a full split / DWT LMS to get to the same MSE level. This demonstrates that the split algorithms are superior than the conventional LMS algorithm in terms of adaptation speed. Furthermore, the convergence behaviour of a full split-path adaptive filter is unified with a DWT adaptation.

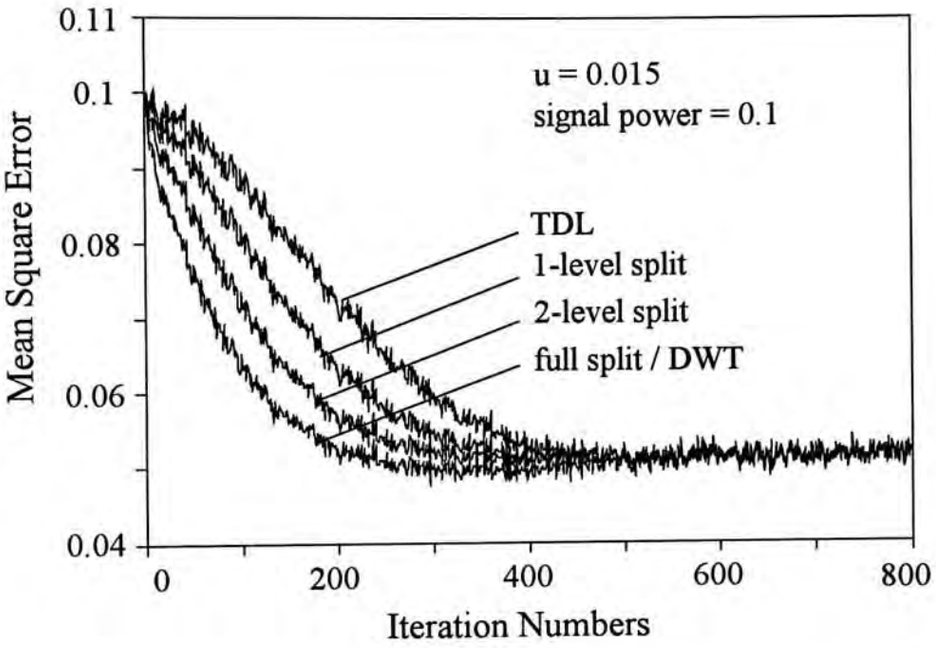


Fig. 6.5. Comparison of learning curves for different adaptive methods.

6.5 Summary

In this chapter, we have established a unified perspective of the multi-stage split-path adaptive filtering method with discrete Walsh transform adaptation. A novel Walsh function is constructed by using the multi-stage splitting matrix. It is shown that the SM defined function is a complete set of Walsh function. The properties of SM-ordered Walsh function is discussed and the conversion rules between SM Walsh function and other typical Walsh functions is developed. In addition, we have shown that the SM-ordered Walsh function provides a useful tool for efficient implementation of the fast discrete Walsh transformation.

An important conclusion can be drawn from the relationship between split-path adaptive filtering and DWT adaptation, that is the split algorithm is essentially a kind of transform or data compression technique. By continuously applying the split operation to the data sequence, the desired information can be merged into a few important coefficients. In the case of adaptive signal processing, the input autocorrelation matrix is diagonalized and the "dominant" information will be located at the upper left corner, which results in a lower eigenvalue spreads distribution. The multi-stage split technique provides a powerful tool for the analysis of DWT adaptation and it is very easy to construct a fast DWT algorithm by means of splitting matrix. In practice, the level of split operation can be selected flexibly according to the requirements of the systems.

Chapter 7 CONCLUSION

In the past two decades, the applications of adaptive signal processing have grown rapidly in expanse of areas, such as linear prediction, noise and echo cancellers, adaptive equalisation, spectrum estimation, adaptive beamforming, system identification and control. Motivated by the increasingly practical need, the theoretical problems associated with adaptive signal processing have been pursued with unprecedented intellectual vigour and numerous literatures have been published. An exhaustive study have been made on the fundamental properties of adaptive algorithms such as speed of convergence, stability, numerical complexity, misadjustment error, robustness, and round-off error analysis. Much of the recent adaptive algorithmic research has focused on algorithms that are based on the gradient-searched method especially the LMS algorithm, since it is the simplest and the most widely used algorithm although it is suffered from relatively slow convergence rate.

To overcome the drawback of the LMS algorithm, split-path adaptive filtering is shown to be an effective approach. When an adaptive filter is split into two linear phase subfilters that are connected in parallel, one antisymmetric while the other symmetric, the eigenvalue of the input signal are divided into two sets, each corresponds to an individual filter path. Due to the partition of eigenvalues, the eigenvalue spread of the two linear phase filters is decreased as well and thus improving the performance of the system.

The major work of this research is concerned with following subjects. One is to resolve the restriction of the "protocol' split-path approach and explore the potential

of extending the split structure to a more general area. Second is to study the properties and performance of transform domain adaptive system when it is configured in a split-path structure. Finally, we wish to find the inherent connection between split-path algorithm and some other transform domain adaptations.

We have first proposed a split-path median LMS (SPMLMS) adaptive algorithm. It is shown that the proposed algorithm not only has a fast convergence speed, but is also effective in suppressing the sparse impulsive noises and in tracking the sharp edges of the desired information.

By analysing the parameter redundancy property of a symmetric/antisymmetric system, we have shown that any $M=2^L$-coefficient system can be decomposed, step by step, into a collection of M parallel single parameter subfilters by L split operations. Therefore, we have developed a generalized modular form of multi-stage split-path structure for the adaptive systems. The essential function of the split technique is its capability for diagonalizing the input autocorrelation matrix thereby reducing the eigenvalue spreads associated with subsystems, which makes the split algorithm a powerful tool to enhance the performance of adaptive filtering. Besides, the multi-stage split algorithm is very easy to implement.

Parallel to the study in time domain, we have also investigated the performance of split-path adaptive algorithm transform domain. We have verified the consistent optimum Wiener solution for split-path, nonsplit-path configurations both in time domain and transform domain adaptive system. Two realization methods of transform domain split-path adaptive filter have been discussed. A practical tracking optimal convergence factor method is derived to speed up the convergence rate of

gradient-based algorithms in a changing environment. It is also shown that the proposed method is less sensitive to the change of system parameter and has a superior robust performance than that of self-orthogonalizing method.

In the thesis, we have presented a new definition of Walsh function by means of continuous splitting matrix. This is an important achievement, not only because the new definition gives a complete set of Walsh function, gives a determined conversion rule between other defined Walsh functions, but also provides a straightforward method for fast DWT algorithm and establishes a unified perspective between split-path adaptive filtering and DWT adaptation.

In the thesis, the analysis and mathematics derivation are rigorous, the problem of applications are emphasised as well. Moreover, all theoretical results are verified and illustrated by extensive computer simulations. It is anticipated that the results of this research will be of great significance in adaptive signal processing.

Besides the works that have been done in this thesis, some other related problems can be studied for further investigation. These include the numerical comparison between different transforms when applied to different applications; fast split-path algorithms for other transform domain adaptation (except for fast DWT, which is discussed in this thesis); and a more precise description of SM-ordered Walsh function (not only for engineering purpose, but as a pure mathematics theory).

REFERENCES

[1] Åstron, K.J. and Wittenmark, B. [1984], *Computer Controlled Systems*, Englewood Cliffs, NJ, Prentice-Hall, 1984.

[2] Ahmed, N. Natarajan, T. and Rao, K.R. [1974], "Discrete cosine transform," *IEEE Trans. Comput.*, vol. C-23, pp. 90-93, Jan. 1974.

[3] Applebaum, S.P. [1976], "Adaptive Arrays," *IEEE Trans. Antennas and Propagation*, vol. AP-24, no. 5, pp. 585-598, September 1976.

[4] Beauchamp, K.G. [1984], *Applications of Walsh and Related Functions, with an Introduction to Sequency Theory*, Academic Press, 1984.

[5] Bershad, N.J. and Feintuch, P.L. [1979], "Analysis of the frequency domain adaptive filter," *Proc. IEEE*, vol. 67, pp. 1658-1659, Dec. 1979.

[6] Bershad, N.J. [1987], "On the optimum gain parameter in LMS adaptation," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-35, no. 7, pp. 1065-1068, July, 1987.

[7] Bershad, N.J. [1989], "Nonlinear quantization effects in the LMS and block LMS adaptive algorithms - a comparison," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-37, no. 10, pp. 1504-1512, October, 1989.

[8] Bitmead, R.P. and Anderson, B.D.O. [1981], "Adaptive frequency sampling filters," *IEEE Trans. Circuits and Systems*, vol. CAS-28, pp. 524-534, June 1981.

[9] Bondyopadhyay, Probir. K. [1988], "Application of running Hartley transform in adaptive digital filtering," *Proc. IEEE*, vol. 76, No. 10, October, 1988.

[10] Bovik, A.C. Huang, T.S. and Munson, D.C [1983], "A generalization of median filtering using linear combinations of order statistics," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-31, pp. 1342-1349, 1983.

[11] Brennan, L.E., Mallet, J.D., and Reed, I.S. [1974], "Rapid convergence rate in adaptive arrays," *IEEE Trans. Aerosp. Electron. Syst.*, AES-10, pp. 853, 1974.

[12] Caraiscos, C. and Liu, B. [1984], "A roundoff error analysis of the LMS adaptive algorithm," *IEEE Trans. Acoustics, Speech, and Signal Processing*, vol. ASSP-32, no. 1, pp. 34-41, February 1984.

[13] Carayannis, G., Manolakis, D.G., and Kalouptsidis, N. [1983], "A fast sequential algorithm for least-squares filtering and prediction," *IEEE Trans. Acoustics, Speech, and Signal Processing*, vol. ASSP-31, no. 6, pp. 1394-1420, December 1983.

[14] Chang, R.W. [1971], "A new equalizer structure for fast start -up digital communication," *Bell System Technical Journal*, vol. 50, pp. 1969-2014, 1971.

[15] Chien, T.M. [1975], "On representation of Walsh functions," *IEEE Trans. Electromag. Compat.*, vol. EMC-17, pp. 170-177, 1975.

[16] Ching, P.C. and Ho, K.C. [1991], "A split structure for adaptive line enhancer," *Int. J. Electronics*, vol. 7, pp. 565-571, Mar. 1991.

[17] Ching, P.C. and Wan, K.F. [1995], "A unified approach to split structure adaptive filtering," *Proceedings, IEEE International Symposium on Circuits and Systems*, Seattle, U.S.A., vol. 3, pp. 1604-1607, April, 1995.

[18] Ciciora, W., Sgrignoli, G., and Thomas, W. [1979], "A tutorial on ghost canceling in television systems," *IEEE Trans. Consum. Electron.*, vol. CE-25, pp. 9, 1979.

[19] Clark, G.A., Mitra, S.K., and Parker, S.R. [1981], "Block implementation of adaptive digital filters," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-29, no. 3, pp. 744-752, June, 1981.

[20] Clarkson, P.M. and Haweel, T.I. [1989], "A median LMS algorithm," *Electron. Lett.* vol. 25, pp. 520-522, 1989.

[21] Delsarte, P. and Genin, Y. [1987], "On the splitting of classical algorithms in linear prediction theory," *IEEE Tans. Acout., Speech, Signal Processing*, vol. ASSP-35, pp. 645-653, May 1987.

[22] Dentino, M., McCool, J. and Widrow, B. [1978], "Adaptive filtering in the frequency domain," *Proc. IEEE*, vol. 66, pp. 1658-1659, Dec. 1978.

[23] Falconer, D.D. and Ljung, L. [1978], "Application of fast Kalman estimation to adaptive equalization," *IEEE Trans. Communication*, vol. COM-26, no. 10, pp. 1439-1446, October 1978.

[24] Farden, D.C. [1981], "Tracking properties of adaptive signal processing algorithms," *IEEE Trans. Acoust. Speech Signal Processing*, vol. ASSP-29, p.439, June 1981.

[25] Ferrara, E.R. [1980], "Fast implementation of LMS adaptive filters," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-28, pp. 474-475, Aug. 1980.

[26] Friedlander, B. [1981], "Lattice filters for adaptive processing," *Proc. IEEE*, vol. 70, no. 8, pp. 829-867, August 1981.

[27] Friedlander, B. [1982], "System identification techniques for adaptive signal processing," *IEEE Trans. Acoustics, Speech and Signal Processing*, vol. 1, no. 1, pp. 699-709, October 1982.

[28] Frost III, O.L. [1972], "An algorithm for linearly constrained adaptive array processing," *Proc. IEEE*, vol. 60, no. 8, pp. 926-935, August 1972.

[29] Gallagher, Jr., N.C. and Wise, G.L. [1981], "A theoretical analysis of the properties of median filters," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-29, pp. 1136-1141, 1981.

[30] Gersho, A. [1969], "Adaptive equalization of highly dispersive channels for data transmission," *The Bell System Technical Journal*, vol. 48, no. 1, pp. 55-70, January 1969.

[31] Gibbs, J.E. [1970], "Discrete complex Walsh transforms," *Proc. Symp. Applic. Walsh Functions*, Washington, D.C., AD707431, pp. 106-122, 1970.

[32] Gitlin, R.D. and Magee, Jr., F.R. [1977], "Self-orthogonalization adaptive equalization algorithms," *IEEE Trans. Communication*, vol. COM-25, no. 7, pp. 666-672, July 1977.

[33] Godard, D. [1974], "Channel equalization using a Kalman filter for fast data transmission," *IBM Journal of Research and Development*, vol. 18, pp. 267-273, May 1974.

[34] Goodwin, G.C. and Payne, R.L. [1977], *Dynamic System Identification, Experimental Design and Data Analysis*, New York, Academic, 1977.

[35] Gray, A.H. and Markel, J.D. [[1975], "A normalized digital filter structure," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-23, pp. 268-277, 1975.

[36] Gray, R. M. [1972], "On the asymptotic eigenvalue distribution of Toeplitz matrices," *IEEE Trans. Inform. Theory*, vol. IT-18, pp. 725-730, Nov. 1972.

[37] Harris, R.W., Chabries, D.M., and Bishop, F.A. [1986], "A variable step (VS) adaptive filter algorithm," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-34, no. 2, pp. 309-316, April, 1986.

[38] Harrison, W.A., Lim, J.S., and Singer, E. [1986], "A new application of adaptive noise cancellations," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-34. pp. 21, 1986.

[39] Haskew, J.R., Kelly, J.M., Kelly, Jr., R.M., and McKinney, T.H. [1973], "Results of a study of the linear prediction vocoder," *IEEE Trans. Commun.*, vol. COM-21, pp. 1008-1014, September 1973.

[40] Haweel, T.I. and Clarkson, P.M. [1992], "A class of order statistic LMS algorithms," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 40, pp. 44-53, 1992.

[41] Haykin, S. [1980], "Array processing application to radar," Stroudsburg, PA: Dowden, Hutchinson & Ross, Inc., 980.

[42] Heideman, M.T. [1992], "Computation of an odd-length DCT from a real-valued DFT of the same length," *IEEE Trans. Signal Processing*, vol. 40, no. 1, January 1992.

[43] Ho, K.C. and Ching, P.C. [1991a], "System identification using a pair of linear phase filter," *Proc. IEEE International Symposium on Circuits and Systems*, pp. 552-555, Singapore, June 1991.

[44] Ho, K.C. and Ching, P.C. [1992a], "Performance analysis of a split-path LMS adaptive filter for AR modeling," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-40, pp. 1375-1382, June 1992.

[45] Ho, K.C. [1991b], "Split algorithms for LMS adaptive systems," Ph.D. thesis, The Chinese University of Hong Kong, May 1991

[46] Ho, K.C., Ching, P.C. and Wan, K.F. [1992b], "Transform domain LMS adaptation of a split-path filter," *Proceedings, IEEE International Symposium on Circuits and Systems,* San Diego, U.S.A., vol. 3, pp. 1597-1600, May 1992.

[47] Hodgkiss, W.S. and Presley, J.A. [1981], "Adaptive tracking of multiple sinusoids whose power levels are widely separated," *IEEE Trans. Circuits and Systems*, vol. CAS-28, no. 6, pp. 550-561, June 1981.

[48] Itakura, F. and Satio, S. [1971], "Digital filtering techniques for speech analysis and synthesis," in Proc. 7th Int. Congr. Acoust. (Budapest, Hungary), 1971.

[49] Jain, A.K. [1974], "A fast Karhunen-Loeve transform for finite discrete images," *Proc. Nat. Electron. Conf.*, Chicago IL, Oct. 1974.

[50] Jain, A.K. [1976], "A fast Karhunen-Loeve transform for a class of random processes," *IEEE Trans. Commun.*, vol. COM-24, pp. 1023-1029, Sept. 1976.

[51] Johnson, C.R. [1988], *Lectures on Adaptive Parameter Estimation*. Englewood Cliffs, NJ: Prentice-Hall, 1988.

[52] Kailath, T. [1974], "A view of three decades of linear filtering theory," *IEEE Trans. Info. Theory*, vol. IT-20, pp. 146, 1974.

[53] Kim, J. and Davisson, L.D. [1975], "Adaptive linear estimation for stationary M-dependent processes," *IEEE Trans. Inform. Theory*, vol. IT-21, pp. 23-31, 1975.

[54] Kitajima, H. [1980], "A symmetric cosine transform," *IEEE Trans. Comput.* vol. C-29, no. 4, pp. 317-323, April 1980.

[55] Koford, J. and Groner, G. [1966], "The use of an adaptive threshold element to design a linear optimal pattern classifier," *IEEE Trans. Inform. Theory*, vol. IT-12, pp. 42-50, Jan. 1966.

[56] Kwong, R.H. [1992], "A variable step size LMS algorithm," *IEEE Trans. Signal Processing*, vol. 40, pp. 1633-1642. July 1992.

[57] Lackey, R.B. and Meltzer, D. [1971], "A simplified definition of the Walsh functions," *IEEE Trans. Comput.*, vol. C-20, pp. 211-213, 1971.

[58] Lee, J.C. and Un, C.K. [1986a], "Block realization of multirate adaptive digital filters," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-34, no. 1, pp. 105-117, Feb. 1986.

[59] Lee, J.C. and Un, C.K. [1986b], "Performance of transform-domain LMS adaptive digital filters," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-34, June 1986.

[60] Ljung, S. and Ljung, L. [1985], "Error propagation properties of recursive least-square adaptation algorithms," *Automatica*, vol. 21, no. 2, 1985.

[61] Lucky, R.W. [1966], "Techniques for adaptive equalization of digital communication systems," *The Bell System Technical Journal*, vol. 45, no. 2, pp. 255-286, February 1966.

[62] Müller, G.S. and Pauw, C.K. [1986], "Acoustic noise cancellation," *Proc. Int. Conf. Acoust., Speech, Signal Processing*, Tokyo, pp. 913, 1986.

[63] Macchi, O. and Eweda, E. [1983], "Second-order convergence analysis of stochastic adaptive linear filtering," *IEEE Trans. Automatic Control*, vol. AC-28, no. 1, pp. 76- 85, January 1983.

[64] Mahalanobis, A., Song, S., Mitra, S.K., and Petraglia, M.R. [1993], "Adaptive FIR filters based on structural subband decomposition for system identification problems," *IEEE Trans. Circuits & Syst.*, vol. 40, no. 6, pp. 375-381, June, 1993.

[65] Makhoul, J. [1973], "Spectral analysis of speech by linear prediction," *IEEE Trans. Audio Electroacoust.*, vol. AU-21, pp. 140-148, June 1973.

[66] Makhoul, J. [1977], "Stable and efficient lattice methods for linear prediction," *IEEE Trans. Acoustics, Speech, and Signal Processing*, vol. ASSP-25, pp. 423-428, October 1977.

[67] Marshall, D.F., *et al.* [1989], "The use of orthogonal transforms for improving performance of adaptive filters," *IEEE Trans. Circuits and Systems*, vol.36, pp.474-484, April, 1989.

[68] Mcwhirter, J.G. and Shepherd, T.J. [1983], "Least-squares lattice algorithm for adaptive channel equalization- a simplified derivation," *IEE Proc.*, vol. 130, F, pp. 532, 1983.

[69] Meckelburg, H.J. and Lipka, D. [1985], "Fast Hartley transform algorithm," *Electronics. Letters*, vol. 21, no. 8, pp. 341-343, April 1985.

[70] Messerschmitt, D.G. [1984], "Echo cancellation in speech and data transmission," *IEEE Journal on Selected Areas in Communication*, vol. SAC-2, no. 2, 00. 283-297, March 1984.

[71] Mitra, S.K., Mahalanobis, A., and Saramäki, T. [1993], "A generalized structural subband decomposition of FIR filters and its application in efficient FIR filter design and implementation," *IEEE Trans. Circuits & Syst.*, vol. 40, no. 6, pp. 363-374, June, 1993.

[72] Narayan, S.S., and Peterson, A.M., [1981], "Frequency domain least-mean-square algorithm," *Proc. IEEE.* vol. 69, pp. 124-126, Jan. 1981.

[73] Narayan, S.S., Peterson, A.M., and Narasimha, M.J. [1983], "Transform domain LMS algorithm," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-31, pp. 609-615, June 1983.

[74] Orfanidis, S.J. [1988], *Optimum Signal Processing: An Introduction*, Second edition, McGraw-Hall, pp. 404-493, 1988.

[75] Patraglia, M.R. and Mitra, S.K. [1993], "Adaptive FIR filter structure based on the generalized subband decomposition of FIR filters," *IEEE Trans. Circuits & Syst.*, vol. 40, no. 6, pp. 354-362, June, 1993.

[76] Pratt, W. [1978], *Digital Image Processing*, New York: Wiley, 1978.

[77] Rabiner, L.R., Sambur, M.R. and Schmidt, C.E. [1975], "Application of a nonlinear smoothing algorithm to speech processing," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-23, pp. 552-557, 1975

[78] Reed, F.A. and Feintuch, P.L. [1981], "A comparison of LMS adaptive cancellers implemented in the frequency domain and the time domain," *IEEE Trans. Circuits and Systems*, vol. CAS-28, pp. 610-615, June 1981.

[79] Restrepo, A. and Bovik, A. [1988], "Adaptive trimmed-mean filters for image restoration," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 36, pp. 1326-1337, 1988.

[80] Riegler, R. and Compton, Jr., R. [1973], "An adaptive array for interference rejection," *Proc. IEEE*, vol. 61, pp. 748-758, June 1973.

[81] Roux, J.Le, and Gueguen, C. [1977], "A fixed point computation of partial correlation coefficients," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-25, pp. 257-259, 1977.

[82] Rushforth, C.K. [1969], "Fast Fourier-Hadamard decoding of orthogonal code," *Information Control*, vol. 15, pp. 33-37, 1969.

[83] Satorius, E.H. and Pack, J.D. [1981], "Application of least-squares lattice algorithms to adaptive equalization," *IEEE Trans. Communications*, vol. COM-29, no. 2, pp. 136-142, February 1981.

[84] Shensa, M.J. [1981], "Recursive least square lattice algorithms: a geometrical approach," *IEEE Trans. Automatic Control*, vol. AC-26, no. 3, pp. 695-702, June 1981.

[85] Sibul, L.H. [1987], *Adaptive Signal Processing*, IEEE Press selected reprint series, New York, 1987.

[86] Sohie, G.R.L. and Sibul, L.H. [1984], "Stochastic convergence properties of the adaptive gradient lattice," *IEEE Trans. Acoustics, Speech, and Signal Processing*, vol. ASSP-32, no. 1, pp. 102-107, February 1984.

[87] Treichler, J.R. [1979], "Transient and convergent behaviour of the adaptive line enhancer," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-27, pp. 53-62, 1979.

[88] Tukey, J.W. [1977], *Exploratory Data Analysis*. Reading, MA: Addison-Wesley, 1977.

[89] Ungerboeck, G. [1972], "Theory on the speed of convergence in adaptive equalizers for digital communication," *IBM Journal of Research and Development*, November 1972.

[90] Waltzan, T. and Schwartz, M. [1973], "Automatic equalization using the discrete frequency domain," *IEEE Trans. Inform. Theory*, vol. IT-19, pp. 59-68, Jan. 1973.

[91] Wan, K.F. and Ching, P.C. [1993], "An adaptive split-path median LMS line enhancer," *Proceedings, International Conference on Neural Networks and Signal Processing*, Guangzhou, China, pp. 705-710, November, 1993.

[92] Wan, K.F. and Ching, P.C. [1994a], "On the optimality of convergence behaviour for transform-domain split-path adaptive filter," *Proceedings, IEEE International Conference on Acoustics, Speech & Signal Processing*, Adelaide, Australia, vol. 3, pp. 421-424, April, 1994.

[93] Wan, K.F. and Ching, P.C. [1994b], "A fast convergence median LMS algorithm," *Proceedings, IEEE International Symposium on Circuits and Systems*, London, UK, vol. 2, PP. 377-380, May, 1994.

[94] Wan, K.F. and Ching, P.C. [1994c], "A fast response split-path median LMS algorithm," *IEEE Trans. Circuits and Systems,* (paper accepted, to be published in 1995)

[95] Wan, K.F. and Ching, P.C. [1995a], "Generalized split structure adaptive algorithm," submitted to *IEEE Signal Processing Letters,* 1995.

[96] Wan, K.F. and Ching, P.C. [1995b], "Performance analysis of split-path adaptive algorithms in transform-domain," submitted to *IEE Proceedings, Vision, Image and Signal Processing,* 1995.

[97] Wan, K.F. and Ching, P.C. [1995c], "Multi-level split-path adaptive filtering and its unification with discrete Walsh transform adaptation," submitted to *IEEE Trans. Circuits and Systems* II, 1995.

[98] Wan, K.F. Ching, P.C. and Ho, K.C. [1992], "DWT domain split-path structure LMS adaptive filter," *Electronics Letters*, vol. 28, pp. 1929-1930, September, 1992.

[99] Wang, Z.D. [1982], "A fast algorithm for the discrete sine transform implemented by the fast cosine transform," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-30, no. 5, pp. 814-815, October 1982.

[100] Watanabe, S. [1965], "Karhunen-Loeve expansion and factor analysis, theoretical remarks and applications," *Trans. 4th Prague Conf. Inform. Theory, Statist. Decision Functions and Random Processes*, Prague, The Netherlands, pp. 635-660, 1965.

[101] Widrow, B. and Hoff, Jr., M. [1960], "Adaptive switching circuits," in *IRE WESCON Conv. Rec.*, pt. 4, pp. 96-104, 1960.

[102] Widrow, B. and Stearns, S.D. [1985], *Adaptive Signal Processing*, Englewood Cliffs, NJ, Prentice-Hall, 1985.

[103] Widrow, B. and Walach, E. [1984], "On the statistical efficiency of the LMS algorithm with nonstationary inputs," *IEEE Trans. Inform. Theory,* vol. IT-30, pp. 211-221, Mar. 1984.

[104] Widrow, B., Glover, Jr., J.R., McCool, J.M., Kaunitz, J., Williams, C.S., Hearn, R.H., Zeidler, J.R., Dong, Jr., E., and Goodlin, R.C. [1975], "Adaptive noise canceling: principles and applications," *Proc. IEEE*, vol. 63, no. 12, pp. 1692-1716, December 1975.

[105] Widrow, B., Mantey, P.E., Griffiths, L.J. and Goode, B.B. [1967], "Adaptive antenna systems," *Proc. IEEE*, vol. 55, no. 12, pp. 2143-2159, December 1967.

[106] Widrow, B., Mantey, P.E., Griffiths, L.J. and Goode, B.B. [1971], "Adaptive filters," in *Aspects of Network and System Theory*, R. Kalman and N. DeClaris, Eds. New York: Holt, Rinehart, and Winston, pp. 563-587. 1971

[107] Widrow, B., McCool, J.M., Larimore, M.G., and Johnson, Jr., C.R. [1976], "Stationary and nonstationary learning characteristics of the LMS adaptive filter," *Proc. IEEE*, vol. 64, no. 8, pp. 1151-1162, August 1976.

[108] Wiener, N. [1949], *Extrapolation, Interpolation and Smoothing of Stationary Time Series with Engineering Applications*, New York, Wiley, 1949.

[109] Williamson, G.A., Clarkson, P.M., and Sethares, W.A. [1993], "Performance characteristics of the median LMS adaptive filter," IEEE Trans. on Signal Processing, vol. 41, no. 2, pp. 667-680, 1993.

[110] Yassa, F.F. [1987], "Optimality in the choice of the convergence factor for gradient-based adaptive algorithms," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-35, pp. 48-59, Jan. 1987.

[111] Yip, P. and Rao, K.R. [1980], "On the computation and the effectiveness of discrete sine transform," *Comput. Electon.*, vol. 7, pp. 45-55, 1980.

[112] Zeidler, J.R., *et al.* [1978], "Adaptive enhancement of multiple sinusoids in uncorrelated noise," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-26, pp. 240-254, 1978.